Univerza v Ljubljani
Fakulteta za računalništvo in informatiko ter
Fakulteta za Matematiko in Fiziko

Mirjam Kolar

Lehmerjev algoritem za računanje največjega skupnega delitelja

diplomsko delo
na interdisciplinarnem univerzitetnem študiju
računalništva in matematike

Mentor: prof. dr. Aleksandar Jurišić
Somentor: prof. dr. Roman Drnovšek

Ljubljana 2015

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Delo naj predstavi osnove, potrebne za razumevanje najpomembnejših metod za računanje največjega skupnega delitelja, kot so Evklidov algoritem (tudi razširjena verzija za iskanje multiplikativnega inverza obrnljivih elementov po modulu izbranega naravnega števila) in verižni ulomki. Opravi naj se tudi časovna primerjava različnih metod ter verjetnostna analiza za ugotavljanje porazdelitve kvocientov.
Glavni cilji so:

(a) Lehmerjev algoritem in njegova implementacija,

(b) Knutova analiza,

(c) primerjava različnih algoritmov glede na časovno zahtevnost,

(d) analiza rezultatov, ki omogoči izbiro optimalnih modulov za učinkovito implementacijo.

Literatura:

M. Urlep, Analiza Evklidovega algoritma, seminarska naloga pri predmetu Kriptografija na FMF, UL, 2005.

S. Maksimović: Učinkovita aritmetika v praštevilskih obsegih, diplomsko delo, Ljubljana, Fakulteta za matematiko in fiziko, 2003.

J. Sorenson, An Analysis of Lehmer's Euclidean GCD Algorithm. In *Proc. AMC ISSAC'95 Symp.*, pages 254–258, 1995.

D. E. Knuth, *The Art of Computer Programming: Seminumerical Algorithms*, Vol. 2, Addison-Wesley, Reading, Ass., 2nd. edition, 1981.

A. Menezes, P. van Oorschot and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press (Series on Discrete Mathematics and its Applications), 4th ed, 1999.

Izjava o avtorstvu diplomskega dela

Spodaj podpisana Mirjam Kolar, z vpisno številko **63080497**, sem avtorica diplomskega dela z naslovom:

Lehmerjev algoritem za računanje največjega skupnega delitelja
Lehmer's GCD algorithm

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelala samostojno pod mentorstvom prof. dr. Aleksandra Jurišića in somentorstvom prof. dr. Romana Drnovška,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 20. januarja 2015 Podpis avtorja:

Kazalo

Povzetek

V nalogi si na kratko ogledamo pravila v zvezi z največjim skupnim deliteljem ter najmanjšim skupnim večkratnikom dveh celih števil in opišemo Evklidov algoritem. Ogledamo si povezavo med Evklidovim algoritmom in verižnimi ulomki, Wirsingovo metodo za določanje porazdelitvene funkcije in kako je Knuth z uporabo te funkcije izračunal porazdelitev delnih kvocientov v Evklidovem algoritmu. Iz tega sledi še opis ideje Lehmerjevega algoritma, v čem se razlikuje od Evklidovega algoritma ter kako z njim poiščemo največji skupni delitelj dveh velikih števil in multiplikativni inverz po modulu n za naravno število n. Algoritma ter njuni razširitvi tudi implementiramo ter primerjamo, kdaj in za koliko je kateri od njiju hitrejši. Na koncu še na naključno izbranih številih preverimo, kakšni so kvocienti v Evklidovem algoritmu v praksi.

Ključne besede:
Lehmerjev algoritem, Evklidov algoritem, verižni ulomki, porazdelitvene funkcije, Wirsingova metoda za določanje porazdelitvene funkcije, kvocienti v Evklidovem algoritmu.

Abstract

In this thesis we briefly look at the rules related to the greatest common divisor and the lowest common multiple of two integers and we describe the Euclidean algorithm. We study connections between Euclidean algorithm and continued fractions, Wirsing's method for determining the distribution function and how Knuth calculated the distribution of partial quotients in Euclidean algorithm using this method. Next Lehmer's algorithm is described and how it improves Euclidean algorithm, greatest common divisor and the multiplicative inverse mod n for a natural number n. We implement both Euclidean algorithm and Lehmer's algorithm in order to compare their speed. We also confirm the calculated distributions of partial quotients in Euclidean algorithm through an experiment.

Keywords:
Lehmer's algorithm, Euclidean algorithm, continued fractions, distribution function, Wirsing's method for determining the distribution function, quotients in Euclidean algorithm.

Poglavje 1

Uvod

Evklidov algoritem je verjetno eden izmed najstarejših znanih algoritmov in je že dve tisočletji znan kot učinkovit algoritem za iskanje največjega skupnega delitelja dveh celih števil. Poleg tega ga uporabljamo tudi za iskanje multiplikativnega inverza po modulu n, kjer je $n \in \mathbb{N}$, reševanje diofantskih enačb, iskanje čim bolj točnega racionalnega približka iracionalnega števila pri dani velikosti števca in imenovalca itd. Z algoritmom za dani različni števili izrazimo večje število kot večkratnik drugega števila, povečan za ostanek (ki je nenegativno število, manjše od drugega števila), in nato postopek nadaljujemo z drugim številom in ostankom itd., vse dokler ne pridemo do ostanka 0. Predzadnji ostanek je iskan največji skupni delitelj. Oglejmo si naslednji primer.

Primer 1.1. Iskanje največjega skupnega delitelja števil $A = 1660695$ in $B = 6840$ z Evklidovim algoritmom.

$$\begin{aligned} 1660695 &= 242 \cdot 6840 + 5415 \\ 6840 &= 1 \cdot 5415 + 1425 \\ 5415 &= 3 \cdot 1425 + 1140 \\ 1425 &= 1 \cdot 1140 + \mathbf{285} \\ 1140 &= 4 \cdot 285 + 0 \end{aligned}$$

Z algoritmom smo izračunali, da je največji skupni delitelj števil 1660695 in 6840 število 285. ♢

Čeprav se v tem primeru zdi, da je računanje kvocientov enostavno, se s povečevanjem števil izkaže, da je to časovno zelo zamudna operacija. Zaradi tega pri velikih številih potrebujemo hitrejšo rešitev. Ena izmed njih je Lehmerjev algoritem, kjer kvociente računamo s pomočjo manjših števil. V veliko primerih namreč lahko izračunamo kvociente tudi, če številom odrežemo zadnjih nekaj števk. Oglejmo si na primeru, kako to storimo.

Primer 1.2. $A = 10\,230\,724\,535\,829\,006\,630, \quad B = 3\,072\,467\,167\,981\,905\,265.$

$$a_0 = \left\lfloor \frac{A}{100\,000\,000\,000\,000} \right\rfloor = 102\,307, \quad a_1 = \left\lfloor \frac{B}{100\,000\,000\,000\,000} \right\rfloor = 30\,724,$$

$$q' = \left\lfloor \frac{a_0 + 1}{a_1} \right\rfloor = \left\lfloor \frac{102\,308}{30\,724} \right\rfloor = 3, \quad q'' = \left\lfloor \frac{a_0}{a_1 + 1} \right\rfloor = \left\lfloor \frac{102\,307}{30\,725} \right\rfloor = 3,$$

$$q' = q'' = 3.$$

Če deljenje preverimo z računalnikom, vidimo, da nam vrne enak rezultat $\left\lfloor \frac{A}{B} \right\rfloor = 3$. Namesto deljenja dveh velikih števil smo prišli do iskanega kvocienta z deljenjem dveh manjših števil. Velja namreč

$$q'' \leq \left\lfloor \frac{A}{B} \right\rfloor \leq q'.$$

♢

Kot bomo videli, se v Evklidovem algoritmu večinoma pojavljajo majhni kvocienti. Kvocienti 1, 2 in 3 se namreč pojavljajo v približno $67,8\,\%$, kar bomo kasneje pokazali tako v teoriji, kot tudi v praksi. Derrick Henry Lehmer je to upošteval v svojem algoritmu, ki kvociente računa s pomočjo manjših števil. Tako je njegov algoritem hitrejši od Evklidovega na velikih številih.

Tak algoritem je predvsem uporaben pri šifriranju, kjer se pogosto uporabljajo velika cela števila. Procesorji namreč ne zmorejo obdelovati tako velikih števil z eno operacijo. Na primer, 32–bitni procesorji lahko računajo s števili do $2^{32} - 1$, ta števila pa so za šifriranje občutno premajhna.

Preden bomo opisali Lehmerjev algoritem, bomo v naslednjih poglavjih na kratko opisali Evklidov algoritem ter s pomočjo verižnih ulomkov in Wirsingove metode za določanje porazdelitvene funkcije pokazali, da je porazdelitev kvocientov res takšna, kot smo zgoraj omenili. Evklidov algoritem je namreč osnova za Lehmerjev algoritem, dejstvo, da se kvocienti res pojavljajo v tolikšnih odstotkih, kot smo omenili in bomo kasneje pokazali, pa nam pove, da je Lehmerjev algoritem res učinkovitejši od Evklidovega za velika števila.

Poglavje 2

Evklidov algoritem

V tem poglavju bomo opisali največji skupni delitelj in najmanjši skupni večkratnik dveh celih števil ter pokazali, kako z Evklidovim algoritmom izračunamo največji skupni delitelj. Ogledali si bomo tudi, kako poiščemo inverz števila v $\mathbb{Z}_n^*$ z razširjenim Evklidovim algoritmom.

2.1 Največji skupni delitelj

Največji skupni delitelj dveh celih števil A in B je največje celo število, ki deli tako A kot tudi B. Označimo ga z $\gcd(A, B)$ (kratica izhaja iz angleškega izraza *greatest common divisor*). Zanj velja:

$$\begin{aligned} \gcd(A, B) &= \gcd(B, A), \\ \gcd(A, B) &= \gcd(-A, B), \\ \gcd(A, 0) &= |A|. \end{aligned}$$

Osnovni izrek aritmetike pravi, da lahko vsako naravno število, ki je večje od 1, zapišemo kot produkt praštevil, pri čemer je ta razcep na praštevila enoličen. Tako lahko števili A in B zapišemo kot

$$\begin{aligned} A &= 2^{a_2} \cdot 3^{a_3} \cdot 5^{a_5} \cdot 7^{a_7} \cdot \ldots = \prod_{p \in \mathbb{P}} p^{a_p}, \\ B &= 2^{b_2} \cdot 3^{b_3} \cdot 5^{b_5} \cdot 7^{b_7} \cdot \ldots = \prod_{p \in \mathbb{P}} p^{b_p}, \end{aligned}$$

pri čemer so eksponenti $a_2, a_3, a_5, a_7, \ldots, b_2, b_3, b_5, b_7, \ldots$ nenegativna cela števila in je le končno mnogo eksponentov različnih od 0. Iz tega zapisa lahko enostavno dobimo formulo za največji skupni delitelj števil A in B, ki je enaka

$$\gcd(A, B) = \prod_{p \in \mathbb{P}} p^{\min(a_p, b_p)} \ . \tag{2.1}$$

Primer 2.1.

$$\begin{aligned} A &= 565950 = 2 \cdot 3 \cdot 5^2 \cdot 7^3 \cdot 11, \\ B &= 780 = 2^2 \cdot 3 \cdot 5 \cdot 13, \\ \gcd(A,B) &= 2 \cdot 3 \cdot 5 = 30. \end{aligned}$$ ♢

Iz definicije (2.1) sledi, da je

$$\gcd(A,B) \cdot C = \gcd(A \cdot C, B \cdot C), \quad \text{za } C \in \mathbb{Z}.$$

Trditev 2.2. *Za največji skupni delitelj velja* $\gcd(A,B) = \gcd(A - q \cdot B, B)$, *za* $q \in \mathbb{Z}$.

Dokaz. Pokažimo, da velja

$$c \,|\, \gcd(A,B) \iff c \,|\, \gcd(A \pm B, B), \quad \text{kjer je } c \in \mathbb{Z}\backslash\{0\}.$$

($\Longrightarrow$) Ker $c \,|\, \gcd(A,B)$, sledi, da je $A = k \cdot c$ in $B = \ell \cdot c$ za neka $k, \ell \in \mathbb{Z}$. Zato je $A \pm B = (k \pm \ell) \cdot c$, natanko tedaj, ko $c \,|\, \gcd(A \pm B, B)$.

($\Longleftarrow$) Naj bo $D = A \pm B$ oziroma $A = D \mp B$. Potem je potrebno pokazati, da iz $c \,|\, \gcd(D,B)$ sledi c$\,|\, \gcd(D \mp B, B)$, kar smo že pokazali v prvem delu.

Tako smo se prepričali, da velja $\gcd(A,B) = \gcd(A \pm B, B)$. Če število B odštejemo oziroma prištejemo večkrat, pridemo do tega, kar smo želeli pokazati. □

Če je $\gcd(A,B) = 1$, pravimo, da sta si števili A in B *tuji*.

2.2 Najmanjši skupni večkratnik

Najmanjši skupni večkratnik dveh celih števil A in B je najmanjše celo pozitivno število, ki je deljivo tako z A kot tudi z B. Označimo ga z $\mathrm{lcm}(A,B)$ (kratica izhaja iz angleškega izraza *lowest common multiple*). Zanj velja:

$$\begin{aligned} \mathrm{lcm}(A,B) &= \mathrm{lcm}(B,A), \\ \mathrm{lcm}(A,B) &= \mathrm{lcm}(-A,B), \\ \mathrm{lcm}(A,0) &= |A|. \end{aligned}$$

Če števili A in B zapišemo kot produkta praštevil, je njun najmanjši skupni večkratnik enak

$$\mathrm{lcm}(A,B) = \prod_{p \in \mathbb{P}} p^{\max(a_p, b_p)} \ . \tag{2.2}$$

Primer 2.3.

$$\begin{aligned} A &= 565950 = 2 \cdot 3 \cdot 5^2 \cdot 7^3 \cdot 11, \\ B &= 780 = 2^2 \cdot 3 \cdot 5 \cdot 13, \\ \operatorname{lcm}(A,B) &= 2^2 \cdot 3 \cdot 5^2 \cdot 7^3 \cdot 11 \cdot 13 = 14714700. \end{aligned}$$ ♢

Iz definicije (2.2) sledi, da je

$$\operatorname{lcm}(A,B) \cdot C = \operatorname{lcm}(A \cdot C, B \cdot C), \quad \text{za } C \in \mathbb{Z}.$$

Če združimo definiciji (2.1) in (2.2), dobimo

$$\gcd(A,B) \cdot \operatorname{lcm}(A,B) = A \cdot B.$$

2.3 Iskanje največjega skupnega delitelja

Enačba (2.1) sicer izgleda uporabna v teoriji, vendar je v praksi problem, če imamo opravka z večjimi števili, saj moramo najprej faktorizirati števili A in B, da lahko potem poiščemo njun največji skupni delitelj. Za faktorizacijo trenutno ne poznamo nobene hitre metode, vseeno pa obstaja učinkovit enostaven algoritem za računanje največjega skupnega delitelja, ki ga je 300 let pr. n. št. opisal Evklid v svojem delu Elementi, v knjigi 7. Ta algoritem je verjetno najstarejši in najpomembnejši algoritem v teoriji števil.
Predpostavimo skozi vso nalogo, da je $A \geq B$. Pri iskanju največjega skupnega delitelja števil A in B algoritem izvaja zaporedje korakov, pri čemer se rezultat vsakega koraka uporabi kot vhodni podatek za naslednji korak.
Definirajmo na začetku kvocient q_0 in ostanek r_0:

$$q_0 = \left\lfloor \frac{A}{B} \right\rfloor, \quad r_0 = A - q_0 \cdot B.$$

Zapišimo definicijo za r_0 drugače:

$$A = q_0 \cdot B + r_0.$$

Iz definicije števil q_0 in r_0 je očitno, da je $r_0 < B$. Ker po trditvi 2.2 velja $\gcd(A,B) = \gcd(q_0 \cdot B + r_0, B) = \gcd(r_0, B)$, lahko problem iskanja $\gcd(A,B)$ prevedemo na problem iskanja $\gcd(B, r_0)$, ki je manjši.
Če zapišemo $q_1 = \left\lfloor \frac{B}{r_0} \right\rfloor$, $r_1 = B - q_1 \cdot r_0$, torej $B = q_1 \cdot r_0 + r_1$, dobimo tako prva dva koraka Evklidovega algoritma. Postopek rekurzivno ponavljamo. Vsak korak se začne z

dvema nenegativnima ostankoma r_{k-1} in r_{k-2}. Cilj k–tega koraka je najti kvocient q_k in ostanek r_k, da velja

$$r_{k-2} = q_k \cdot r_{k-1} + r_k, \quad \text{kjer je } r_k < r_{k-1}. \tag{2.3}$$

V prvem koraku ($k = 0$) velja $A = r_{-2}$, $B = r_{-1}$. Prvih nekaj korakov algoritma je

$$\begin{aligned} A &= q_0 \cdot B + r_0, \\ B &= q_1 \cdot r_0 + r_1, \\ r_0 &= q_2 \cdot r_1 + r_2, \\ r_1 &= q_3 \cdot r_2 + r_3, \\ &\vdots \end{aligned}$$

Zaporedje $\{r_i\}$, $i = -2, -1, \ldots$, imenujemo *Evklidovo zaporedje* števil A in B.

Trditev 2.4. *Evklidov algoritem se konča po končnem številu korakov in vrne pravilen rezultat.*

Dokaz. Zaporedje $\{r_i\}$ nenegativnih celih števil je strogo padajoče, kar pomeni, da po končnem številu korakov pridemo do 0, tj. $r_N = 0$ za nek $N \in \mathbb{N}$.
Po trditvi 2.2 velja:

$$\begin{aligned} \gcd(A, B) = \gcd(B, r_0) &= \gcd(r_0, r_1) = \cdots = \gcd(r_{N-2}, r_{N-1}) \\ &= \gcd(r_{N-1}, r_N) = \gcd(r_{N-1}, 0) = r_{N-1}, \end{aligned}$$

torej nam algoritem res vrne pravilen rezultat, tj. $\gcd(A, B) = r_{N-1}$. □

Oglejmo si še, kolikšno je število korakov v Evklidovem algoritmu. Predpostavimo, da sta števili A in B naključno porazdeljeni med številoma 1 in M. Lamé je pokazal, da je število korakov v Evklidovem algoritmu enako največ

$$\left\lceil \frac{\ln(\sqrt{5}M)}{\ln((1+\sqrt{5})/2)} \right\rceil - 2 \approx 2,078 \cdot \ln M + 1,672,$$

Heilbronn pa, da je povprečno število korakov v Evklidovem algoritmu približno enako

$$\frac{12 \cdot \ln 2}{\pi^2} \cdot \ln M + 0,14 \approx 0,843 \ln M + 0,14.$$

Dokaz najdemo npr. v knjigi The Art of Computer Programming ([6]), poglavje 4.5.3. Analysis of Euclid's Algorithm.

Izrek 2.5. *Za števili $A \geq B$, ki sta naključno porazdeljeni med številoma 1 in M, je časovna zahtevnost Evklidovega algoritma enaka $O((\log_2 M)^2)$.*

Dokaz. Časovna zahtevnost izračuna $q \cdot B + r$ je enaka $O(\log_2 B \cdot \log_2 q)$, saj je časovna zahtevnost množenja dveh celih števil m in n enaka $O(\log_2 m \cdot \log_2 n)$. Zahtevnost $\left\lfloor \frac{A}{B} \right\rfloor$ je enaka $O((\log_2 M)^2)$. Ker je število M večje od števil B in q, je tako časovna zahtevnost enega koraka $O((\log_2 M)^2)$. Ker je število korakov $O(\log_2 M)$, je časovna zahtevnost algoritma enaka $O((\log_2 M)^3)$.
Poskusimo sedaj dokazati, da je časovna zahtevnost manjša. Pokažimo najprej, da se število A po dveh korakih v Evklidovem zaporedju zmanjša vsaj za faktor 2.
Vemo, da velja $r_k < r_{k-1}$. Razdelimo dokaz na dva primera:

(1) $r_{k-1} \leq \dfrac{r_{k-2}}{2} \implies r_k < r_{k-1} \leq \dfrac{r_{k-2}}{2}$,

(2) $r_{k-1} > \dfrac{r_{k-2}}{2} \implies$ po definiciji (2.3) velja $r_k = r_{k-2} - q_k \cdot r_{k-1}$.
Edini možni primer za q_k je $q_k = 1$ in v tem primeru velja

$$r_k = r_{k-2} - r_{k-1} < \frac{r_{k-2}}{2}.$$

Če je v prvi (in morda še drugi) vrstici zahtevnost algoritma $O((\log_2 M)^2)$, je zahtevnost vseh ostalih vrstic tako skupaj manjša od zahtevnosti prve vrstice, torej od $O((\log_2 M)^2)$. Skupaj je tako časovna zahtevnost manjša ali enaka vsoti zahtevnosti prve vrstice, druge vrstice in vseh ostalih vrstic skupaj, kar pomeni, da je manjša ali enaka $O((\log_2 M)^2) + O((\log_2 M)^2) + O((\log_2 M)^2) = 3 \cdot O((\log_2 M)^2) = O((\log_2 M)^2)$. □

Primer računanja Evklidovega algoritma smo si ogledali že v uvodu (Primer 1.1), še enega pa si oglejmo sedaj.

Primer 2.6. $A = 18371$, $B = 329$.

								i	q_i	r_i
								-2		18371
								-1		329
18371	=	55	·	329	+	276		0	55	276
329	=	1	·	276	+	53		1	1	53
276	=	5	·	53	+	11		2	5	11
53	=	4	·	11	+	9		3	4	9
11	=	1	·	9	+	2		4	1	2
9	=	4	·	2	+	**1**		5	4	**1**
2	=	2	·	1	+	0		6	2	0

Števili 18371 in 329 sta si tuji, saj je $\gcd(18371, 329) = 1$. ♢

2.4 Iskanje inverza v $\mathbb{Z}_n^*$

Če Evklidov algoritem razširimo, lahko z njim poiščemo inverz števila v $\mathbb{Z}_n^*$.
Definirajmo poleg Evklidovega zaporedja $\{r_i\}$ še zaporedji $\{s_i\}$ in $\{t_i\}$, s katerima računamo inverz števila v $\mathbb{Z}_n^*$. Zanju velja:

$$\begin{aligned} &s_{-2} = 1, \quad t_{-2} = 0, \\ &s_{-1} = 0, \quad t_{-1} = 1, \\ &s_i \cdot A + t_i \cdot B = r_i, \qquad \text{za } i = -2, -1, \ldots \end{aligned} \tag{2.4}$$

Izrek 2.7. *Z uporabo razširjenega Evklidovega algoritma lahko poiščemo števili x in y, ki sta eni izmed rešitev diofantske enačbe z dvema neznankama*

$$x \cdot A + y \cdot B = C,$$

natanko tedaj, ko $\gcd(A, B)$ *deli* C.

Dokaz. Najbolj enostavna primera sta, ko je $C = A$ ali ko je $C = B$:

$$\begin{aligned} 1 \cdot A + 0 \cdot B &= A, \\ 0 \cdot A + 1 \cdot B &= B. \end{aligned} \tag{2.5}$$

V primeru, ko je $C = \gcd(A, B)$, nam enačba (2.5) omogoči, da iščemo rešitve z zaporedji $\{s_i\}$, $\{t_i\}$ in $\{r_i\}$. Ideja je, da gre r_i dovolj hitro proti $\gcd(A, B)$. Tako je $r_i = r_{i-2} - q_i \cdot r_{i-1}$. Če q_{i+1}–krat odštejemo enačbo (2.4) predhodni enačbi

$$s_{i-1} \cdot A + t_{i-1} \cdot B = r_{i-1},$$

dobimo

$$A(s_{i-1} - q_{i+1} \cdot s_i) + B(t_{i-1} - q_{i+1} \cdot t_i) = r_{i+1}.$$

Zaporedji $\{s_i\}$ in $\{t_i\}$ tako računamo po enakem principu kot zaporedje r_i:

$$\begin{aligned} s_{k-2} &= q_k \cdot s_{k-1} + s_k, \\ t_{k-2} &= q_k \cdot t_{k-1} + t_k. \end{aligned}$$

Ko se algoritem konča z $r_N = 0$, velja $r_{N-1} = \gcd(A, B)$, prav tako velja tudi $s_{N-1} = s$ ter $t_{N-1} = t$.
Števili s in t tako že imamo. Kako sedaj dobimo števili x in y, rešitvi linearne diofantske enačbe? Ker velja $\gcd(A, B) \big| C$, je $C = C' \cdot \gcd(A, B)$. Tako lahko enačbo $s \cdot A + t \cdot B = \gcd(A, B)$ pomnožimo s C' in dobimo $(s \cdot C') \cdot A + (t \cdot C') \cdot B = \gcd(A, B) \cdot C' = C$. Rešitvi linearne diofantske enačbe sta tako

$$x = s \cdot C' \quad \text{in} \quad y = t \cdot C'.$$

□

Rešitvi x in y sta le eni izmed neskončno rešitev diofantske enačbe, s pomočjo teh rešitev pa lahko dobimo vse ostale.

Kot vidimo, pri Evklidovem algoritmu računamo zaporedje $\{r_i\}$, če pa računamo še zaporedje $\{s_i\}$ in/ali zaporedje $\{t_i\}$, imenujemo ta algoritem *razširjen Evklidov algoritem.*

Ko sta si števili A in B tuji, torej ko velja $\gcd(A, B) = 1$, lahko z uporabo razširjenega Evklidovega algoritma poiščemo inverz števila A po modulu B in inverz števila B po modulu A. Število s je v tem primeru multiplikativni inverz števila A po modulu B in število t je multiplikativni inverz števila B po modulu A. Vendar za inverz računamo poleg zaporedja $\{r_i\}$ le $\{s_i\}$ ali le $\{t_i\}$, saj potrebujemo le enega od njiju, odvisno za katero od števil A oziroma B računamo inverz. Tako lahko z razširjenim Evklidovim algoritmom poiščemo multiplikativni inverz obrnljivih elementov po modulu izbranega naravnega števila.

Primer 2.8. Oglejmo si primer računanja inverza na primeru 2.6 iz prejšnjega poglavja za tuji si števili $A = 18371$ in $B = 329$ z uporabo razširjenega Evklidovega algoritma.

Računanje zaporedja s_i:

$$\begin{aligned}
1 &= 55 \cdot 0 + 1 \\
0 &= 1 \cdot 1 + (-1) \\
1 &= 5 \cdot (-1) + 6 \\
-1 &= 4 \cdot 6 + (-25) \\
6 &= 1 \cdot (-25) + 31 \\
-25 &= 4 \cdot 31 + (\mathbf{-149}) \\
31 &= 2 \cdot (-149) + 329
\end{aligned}$$

Računanje zaporedja t_i:

$$\begin{aligned}
0 &= 55 \cdot 1 + (-55) \\
1 &= 1 \cdot (-55) + 56 \\
-55 &= 5 \cdot 56 + (-335) \\
56 &= 4 \cdot (-335) + 1396 \\
-335 &= 1 \cdot 1396 + (-1731) \\
1396 &= 4 \cdot (-1731) + \mathbf{8320} \\
-1731 &= 2 \cdot 8320 + (-18371)
\end{aligned}$$

Zapis s tabelo:

i	q_i	r_i	s_i	t_i
-2		18371	1	0
-1		329	0	1
0	55	276	1	-55
1	1	53	-1	56
2	5	11	6	-335
3	4	9	-25	1396
4	1	2	31	-1731
5	4	**1**	$\mathbf{-149}$	**8320**
6	2	0		

Torej je $18371 \cdot (-149) + 329 \cdot 8320 = 1$.

Preverimo inverza:

$$-149 \equiv 180 \pmod{329}, \quad 18371 \cdot 180 = 3306780 \equiv 1 \pmod{329},$$
$$329 \cdot 8320 = 3306780 \equiv 1 \pmod{18371}.$$ ♢

Poglavje 3

Verižni ulomki

Realno število $a = a_0, a_1a_2a_3\ldots$ lahko predstavimo z zaporedjem $\{a_n\}$:

$$a = a_0 + \frac{a_1}{10} + \frac{a_2}{100} + \frac{a_3}{1000} + \cdots$$

Pri tem je $a_0 \in \mathbb{Z}$, $a_i \in \mathbb{N}$, za $i > 0$. Daljše kot je zaporedje, bolj se približujemo vrednosti števila a. Verižni ulomki so alternativa temu zaporedju, saj lahko realno število predstavimo tudi z njimi. Vendar verižni ulomki niso zanimivi le zaradi tega, ampak tudi, ker so tesno povezani z Evklidovim algoritmom.

Mi se bomo večinoma ukvarjali s končnimi verižnimi ulomki, za katere velja, da so vsi števci enaki 1. Oglejmo si njihov zapis na dva načina:

$$a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{\cdots \cfrac{}{a_{n-1} + \cfrac{1}{a_n}}}}} = a_0 + 1/(a_1 + 1/(a_2 + 1/(\ldots/(a_{n-1} + 1/a_n)\ldots))).$$

Zgornji ulomek na kratko zapišemo v naslednji obliki:

$$[a_0; a_1, a_2, a_3, \ldots, a_n].$$

V primeru, da je $a_0 = 0$, kar bomo skozi nalogo tudi privzeli, izpustimo a_0 in zapišemo kar $[a_1, a_2, a_3, \ldots, a_n]$.

3.1 Povezava Q–polinomov z verižnimi ulomki

V tem razdelku nas bo zanimalo, kako izgleda verižni ulomek, če ga poskusimo izraziti kot običajni ulomek, to je kvocient dveh celih števil. Oglejmo si najkrajše tri primere

verižnih ulomkov:

$$\begin{aligned}
[a_1] &= \frac{1}{a_1}, \\
[a_1, a_2] &= \frac{1}{a_1 + \dfrac{1}{a_2}} = \frac{a_2}{a_1 \cdot a_2 + 1}, \\
[a_1, a_2, a_3] &= \frac{1}{a_1 + \dfrac{1}{a_2 + \dfrac{1}{a_3}}} = \frac{1}{a_1 + \dfrac{a_3}{a_2 \cdot a_3 + 1}} = \frac{a_2 \cdot a_3 + 1}{a_1 \cdot a_2 \cdot a_3 + a_1 + a_3}.
\end{aligned} \tag{3.1}$$

Opazimo, da v števcu in imenovalcu dobimo polinome, kar nas pripelje do Q–polinomov več spremenljivk, ki so definirani z

$$[x_1, x_2, \ldots, x_n] = \frac{Q_{n-1}(x_2, \ldots, x_n)}{Q_n(x_1, x_2, \ldots, x_n)}. \tag{3.2}$$

Iz (3.1) in (3.2) dobimo $Q_0 = 1$ in $Q_1(x_1) = x_1$, nato pa iz rekurzije

$$[x_1, x_2, \ldots, x_n] = \frac{1}{x_1 + [x_2, x_3, \ldots, x_n]}$$

izračunamo

$$\frac{Q_{n-1}(x_2, \ldots, x_n)}{Q_n(x_1, x_2, \ldots, x_n)} = \frac{1}{x_1 + \frac{Q_{n-2}(x_3, \ldots, x_n)}{Q_{n-1}(x_2, \ldots, x_n)}}$$

oziroma

$$Q_n(x_1, x_2, \ldots, x_n) = x_1 \cdot Q_{n-1}(x_2, \ldots, x_n) + Q_{n-2}(x_3, \ldots, x_n). \tag{3.3}$$

Oglejmo si prvih nekaj primerov Q–polinomov:

$$\begin{aligned}
&Q_0 = 1, \\
&Q_1(x_1) = x_1, \\
&Q_2(x_1, x_2) = x_1 \cdot Q_1(x_2) + Q_0 = x_1 \cdot x_2 + 1, \\
&Q_3(x_1, x_2, x_3) = x_1 \cdot Q_2(x_2, x_3) + Q_1(x_3) = x_1 \cdot (x_2 \cdot x_3 + 1) + x_3 \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad = x_1 \cdot x_2 \cdot x_3 + x_1 + x_3.
\end{aligned}$$

Za Q–polinome velja:

$$Q_n(x_1, x_2, \ldots, x_n) = Q_n(x_n, x_{n-1}, \ldots, x_1). \tag{3.4}$$

Kot posledico enačb (3.3) in (3.4) dobimo

$$Q_n(x_1, x_2, \ldots, x_n) = x_n \cdot Q_{n-1}(x_1, \ldots, x_{n-1}) + Q_{n-2}(x_1, x_2, \ldots, x_{n-2}).$$

Trditev 3.1. *Za Q–polinome velja enakost*

$$Q_n(x_1,\ldots,x_n)\cdot Q_n(x_2,\ldots,x_{n+1}) - Q_{n+1}(x_1,\ldots,x_{n+1})\cdot Q_{n-1}(x_2,\ldots,x_n) \;=\; (-1)^n, \quad n\geq 1. \tag{3.5}$$

Dokaz. Uporabimo indukcijo.

$$\begin{aligned}
n=1: \quad & Q_1(x_1)\cdot Q_1(x_2) - Q_2(x_1,x_2)\cdot Q_0 = x_1\cdot x_2 - (x_1\cdot x_2+1) = -1,\\
n\longrightarrow n+1: \quad & Q_{n+1}(x_1,\ldots,x_{n+1})\cdot Q_{n+1}(x_2,\ldots,x_{n+2})\\
& \quad - Q_{n+2}(x_1,\ldots,x_{n+2})\cdot Q_n(x_2,\ldots,x_{n+1})\\
& = Q_{n+1}(x_1,\ldots,x_{n+1})\cdot\big(x_{n+2}\cdot Q_n(x_2,\ldots,x_{n+1}) + Q_{n-1}(x_2,\ldots,x_n)\big)\\
& \quad - \big(x_{n+2}\cdot Q_{n+1}(x_1,\ldots,x_{n+1}) + Q_n(x_1,\ldots,x_n)\big)\cdot Q_n(x_2,\ldots,x_{n+1})\\
& = x_{n+2}\cdot\big(Q_{n+1}(x_1,\ldots,x_{n+1})\cdot Q_n(x_2,\ldots,x_{n+1})\\
& \quad - Q_{n+1}(x_1,\ldots,x_{n+1})\cdot Q_n(x_2,\ldots,x_{n+1})\big) - (-1)^n = (-1)^{n+1}\,.
\end{aligned}$$

□

Trditev 3.2. *Če označimo $q_k = Q_k(x_1,\ldots,x_k)$, velja*

$$[x_1,\ldots,x_n] = \frac{1}{q_0\cdot q_1} - \frac{1}{q_1\cdot q_2} + \frac{1}{q_2\cdot q_3} \mp \cdots + \frac{(-1)^{n-1}}{q_{n-1}\cdot q_n}.$$

Dokaz. Uporabimo indukcijo.

$n=1:$ $\quad [x_1] = \dfrac{1}{Q_0\cdot Q_1(x_1)} = \dfrac{1}{q_0\cdot q_1},$

$n\longrightarrow n+1:$ Če enačbo (3.5) delimo s

$$Q_n(x_1,\ldots,x_n) \text{ in } Q_{n+1}(x_1,\ldots,x_{n+1}),$$

dobimo

$$\frac{Q_n(x_2,\ldots,x_{n+1})}{q_{n+1}} - \frac{Q_{n-1}(x_2,\ldots,x_n)}{q_n} = \frac{(-1)^n}{q_n\cdot q_{n+1}}.$$

Če v to enačbo vstavimo še enakost (3.2), dobimo

$$[x_1,\ldots,x_{n+1}] - [x_1,\ldots,x_n] = \frac{(-1)^n}{q_n\cdot q_{n+1}},$$

od tod pa z indukcijsko predpostavko:

$$[x_1,\ldots,x_{n+1}] = [x_1,\ldots,x_n] + \frac{(-1)^n}{q_n\cdot q_{n+1}} = \frac{1}{q_0\cdot q_1} - \frac{1}{q_1\cdot q_2} \pm \cdots + \frac{(-1)^n}{q_n\cdot q_{n+1}}.$$

□

3.2 Realna števila in verižni ulomki

Oglejmo si, kako lahko povežemo z verižnimi ulomki realna števila.

Vsako realno število $X \in [0, 1)$ lahko zapišemo kot enostavni verižni ulomek. Definirajmo ga takole:

$$\begin{aligned} &X_0 = X, \\ &A_{n+1} = \left\lfloor \frac{1}{X_n} \right\rfloor, \quad X_{n+1} = \frac{1}{X_n} - A_{n+1}, \end{aligned} \tag{3.6}$$

za vsak $n \geq 0$, za katerega velja $X_n \neq 0$.

Števila $A_1, A_2, \ldots$ se imenujejo delni kvocienti števila X. Če je $X_n = 0$, potem nista definirana niti A_{n+1} niti X_{n+1}. Če pa velja $X_n \neq 0$, iz definicije sledi, da je $0 \leq X_{n+1} < 1$ (saj iz (3.6) sledi $X_{n+1} = \frac{1}{X_n} - \left\lfloor \frac{1}{X_n} \right\rfloor$). To pomeni, da je A_n naravno število.

Iz definicije (3.6) lahko izračunamo

$$\begin{aligned} X_{n+1} &= \frac{1}{X_n} - A_{n+1}, \\ X_n &= \frac{1}{A_{n+1} + X_{n+1}}. \end{aligned} \tag{3.7}$$

Iz tega vidimo, da velja

$$X = X_0 = \frac{1}{A_1 + X_1} = \frac{1}{A_1 + \dfrac{1}{A_2 + X_2}} = \cdots,$$

torej je

$$X = [A_1, A_2, \ldots, A_n + X_n] \tag{3.8}$$

za vsak $n \geq 1$, kadarkoli je X_n definiran. Če je $X_n = 0$, je $X = [A_1, A_2, \ldots, A_n]$.

Oglejmo si primer izračuna zapisa realnega števila z verižnim ulomkom.

Primer 3.3. $X = \dfrac{125}{2044}$.

$$
\begin{aligned}
& & X_0 &= X = \frac{125}{2044},\\
A_1 &= \left\lfloor \tfrac{1}{X_0} \right\rfloor = \left\lfloor \tfrac{2044}{125} \right\rfloor = 16, & X_1 &= \frac{1}{X_0} - A_1 = \frac{2044}{125} - 16 = \frac{44}{125},\\
A_2 &= \left\lfloor \tfrac{1}{X_1} \right\rfloor = \left\lfloor \tfrac{125}{44} \right\rfloor = 2, & X_2 &= \frac{1}{X_1} - A_2 = \frac{125}{44} - 2 = \frac{37}{44},\\
A_3 &= \left\lfloor \tfrac{1}{X_2} \right\rfloor = \left\lfloor \tfrac{44}{37} \right\rfloor = 1, & X_3 &= \frac{1}{X_2} - A_3 = \frac{44}{37} - 1 = \frac{7}{37},\\
A_4 &= \left\lfloor \tfrac{1}{X_3} \right\rfloor = \left\lfloor \tfrac{37}{7} \right\rfloor = 5, & X_4 &= \frac{1}{X_3} - A_4 = \frac{37}{7} - 5 = \frac{2}{7},\\
A_5 &= \left\lfloor \tfrac{1}{X_4} \right\rfloor = \left\lfloor \tfrac{7}{2} \right\rfloor = 3, & X_5 &= \frac{1}{X_4} - A_5 = \frac{7}{2} - 3 = \frac{1}{2},\\
A_6 &= \left\lfloor \tfrac{1}{X_5} \right\rfloor = \left\lfloor \tfrac{2}{1} \right\rfloor = 2, & X_6 &= \frac{1}{X_5} - A_6 = \frac{2}{1} - 2 = 0.
\end{aligned}
$$

$X = [16, 2, 1, 5, 3, 2]$. ♢

Trditev 3.4. *Če velja $X_n \neq 0$, potem število X vedno leži med $[A_1, A_2, \ldots, A_n]$ in $[A_1, A_2, \ldots, A_n + 1]$, saj je $X_n < 1$. Natančneje*

$$\big|X - [A_1, A_2, \ldots, A_n]\big| \leq \frac{1}{Q_{n+1}(A_1, \ldots, A_n, A_{n+1}) \cdot Q_n(A_1, \ldots, A_n)}. \tag{3.9}$$

Dokaz. Iz (3.8) dobimo

$$\big|X - [A_1, A_2, \ldots, A_n]\big| = \big|[A_1, A_2, \ldots, A_n + X_n] - [A_1, A_2, \ldots, A_n]\big|.$$

Iz enačbe (3.7) sledi

$$\big|[A_1, A_2, \ldots, A_n + X_n] - [A_1, A_2, \ldots, A_n]\big| = \big|[A_1, A_2, \ldots, A_n, \tfrac{1}{X_n}] - [A_1, A_2, \ldots, A_n]\big|.$$

Iz osnovne lastnosti Q–polinomov (3.2) sledi

$$
\begin{aligned}
\big|[A_1, A_2, \ldots, A_n, \tfrac{1}{X_n}] - [A_1, A_2, \ldots, A_n]\big| &= \left| \frac{Q_n(A_2, \ldots, A_n, \frac{1}{X_n})}{Q_{n+1}(A_1, \ldots, A_n, \frac{1}{X_n})} - \frac{Q_{n-1}(A_2, \ldots, A_n)}{Q_n(A_1, \ldots, A_n)} \right|\\
= \left| \frac{Q_n(A_2, \ldots, A_n, \frac{1}{X_n}) \cdot Q_n(A_1, \ldots, A_n) - Q_{n-1}(A_2, \ldots, A_n) \cdot Q_{n+1}(A_1, \ldots, A_n, \frac{1}{X_n})}{Q_{n+1}(A_1, \ldots, A_n, \frac{1}{X_n}) \cdot Q_n(A_1, \ldots, A_n)} \right|&.
\end{aligned}
$$

Zaradi lastnosti Q–polinomov (3.5) je prejšnja enačba enaka

$$
\begin{aligned}
&\left| \frac{Q_n(A_2, \ldots, A_n, \frac{1}{X_n}) \cdot Q_n(A_1, \ldots, A_n) - Q_{n-1}(A_2, \ldots, A_n) \cdot Q_{n+1}(A_1, \ldots, A_n, \frac{1}{X_n})}{Q_{n+1}(A_1, \ldots, A_n, \frac{1}{X_n}) \cdot Q_n(A_1, \ldots, A_n)} \right|\\
&= \frac{1}{Q_{n+1}(A_1, \ldots, A_n, \frac{1}{X_n}) \cdot Q_n(A_1, \ldots, A_n)} \leq \frac{1}{Q_{n+1}(A_1, \ldots, A_n, A_{n+1}) \cdot Q_n(A_1, \ldots, A_n)}.
\end{aligned}
$$

Zadnji neenačaj velja zaradi tega, ker zaradi definicije (3.6), $A_{n+1} = \left\lfloor \frac{1}{X_n} \right\rfloor$, velja

$$Q_{n+1}(A_1, \ldots, A_n, A_{n+1}) \leq Q_{n+1}(A_1, \ldots, A_n, \tfrac{1}{X_n}).$$

□

Iz enačbe (3.9) vidimo, da ko gre $n \to \infty$, gre vrednost $\left|X - [A_1, A_2, \ldots, A_n]\right|$ proti 0, zato je $[A_1, A_2, \ldots, A_n]$ zelo dobra aproksimacija števila X za velike n.
Če je $X_n = 0$, je število X racionalno, saj ga lahko predstavimo z enostavnim končnim verižnim ulomkom. Če pa je X iracionalno število, je nemogoče, da bi bil $X_n = 0$ za katerikoli n. Zato za iracionalna števila enostavne končne verižne ulomke razširimo na enostavne neskončne verižne ulomke $[A_1, A_2, \ldots]$. Neskončni verižni ulomek definiramo kot

$$[A_1, A_2, \ldots] = \lim_{n\to\infty} [A_1, A_2, \ldots, A_n]$$

in iz (3.9) je očitno, da je limita enaka X.

3.3 Evklidov algoritem in verižni ulomki

Ko je X racionalno število, lahko enostavni verižni ulomek povežemo z Evklidovim algoritmom. V tem poglavju si bomo ogledali, kako.
Recimo, da je $X = \dfrac{B}{A}$, kjer je $0 \leq B < A$. Pri enostavnem verižnem ulomku velja $X_0 = X$. Vzemimo, da je $U_0 = A$ in $V_0 = B$ in predpostavimo, da $X_n = \dfrac{V_n}{U_n} \neq 0$. Potem iz definicije (3.6) sledi, da je

$$\begin{aligned} A_{n+1} &= \left\lfloor \frac{1}{X_n} \right\rfloor = \left\lfloor \frac{U_n}{V_n} \right\rfloor, \\ X_{n+1} &= \frac{1}{X_n} - A_{n+1} = \frac{U_n}{V_n} - \left\lfloor \frac{U_n}{V_n} \right\rfloor = \frac{U_n \bmod V_n}{V_n}. \end{aligned} \tag{3.10}$$

Če vzamemo, da je

$$U_{n+1} = V_n, \quad V_{n+1} = U_n \bmod V_n, \tag{3.11}$$

bo s tako definicijo pogoj $X_n = \dfrac{V_n}{U_n}$ veljal skozi ves postopek.
Oglejmo si postopek iskanja največjega skupnega delitelja, pri tem pa uporabimo število X iz primera (3.3).

Primer 3.5. $X = \dfrac{125}{2044} = [16, 2, 1, 5, 3, 2]$, $A = 2044$, $B = 125$.

Evklidov algoritem:			Izračun z verižnim ulomkom:	
r_{k-2}	$=$	$q_k \cdot r_{k-1} + r_k\,,$		$X_0 = \dfrac{125}{2044},$
2044	$=$	$16 \cdot 125 + 44\,,$	$A_1 = 16,$	$X_1 = \dfrac{44}{125},$
125	$=$	$2 \cdot 44 + 37\,,$	$A_2 = 2,$	$X_2 = \dfrac{37}{44},$
44	$=$	$1 \cdot 37 + 7\,,$	$A_3 = 1,$	$X_3 = \dfrac{7}{37},$
37	$=$	$5 \cdot 7 + 2\,,$	$A_4 = 5,$	$X_4 = \dfrac{2}{7},$
7	$=$	$3 \cdot 2 + \mathbf{1}\,,$	$A_5 = 3,$	$X_5 = \dfrac{\mathbf{1}}{2},$
2	$=$	$2 \cdot 1 + 0\,,$	$A_6 = 2,$	$X_6 = 0.$

$\gcd(2044, 125) = 1$.

Lahko vidimo, da je definicija (3.11) transformacija, narejena na spremenljivkah r_k in r_{k+1} v Evklidovem algoritmu. Za ta primer lahko iz $X = \frac{125}{2044} = [16, 2, 1, 5, 3, 2]$ točno vidimo, da bo Evklidov algoritem za ta primer porabil 6 korakov in da bodo kvocienti $q_k = \left\lfloor \frac{r_{k-2}}{r_{k-1}} \right\rfloor$ zaporedoma enaki $16, 2, 1, 5, 3, 2$. ♢

Trditev 3.6. *Postopek* (3.10) *nam vrne največji skupni delitelj števil A in B.*

Dokaz. V prejšnjem poglavju smo povedali, da X_n ne more biti enak 0, če je X iracionalno število. Za Evklidov algoritem vemo, da se vedno konča. Iz teh dveh lastnosti ter iz povezave med Evklidovim algoritmom in enostavnimi verižnimi ulomki vidimo, da se enostavni verižni ulomek za X konča pri nekem koraku z $X_n = 0$ če in samo če je X racionalno število.

Če so dobljeni kvocienti, ki jih dobimo v Evklidovem algoritmu, enaki $A_1, A_2, \ldots, A_n$, potem je $X_n = 0$ in imamo po (3.2):

$$\frac{B}{A} = [A_1, A_2, \ldots, A_n] = \frac{Q_{n-1}(A_2, \ldots, A_n)}{Q_n(A_1, A_2, \ldots, A_n)}, \qquad n \geq 1. \tag{3.12}$$

Če v enačbi (3.5) n nadomestimo z $n - 1$, dobimo

$$y \cdot Q_{n-1}(x_2, \ldots, x_n) - Q_n(x_1, \ldots, x_n) \cdot z = (-1)^{n-1}, \qquad n \geq 1,$$

kjer sta y in z neki racionalni števili. Iz tega vidimo, da sta $Q_{n-1}(x_2, \ldots, x_n)$ in $Q_n(x_1, \ldots, x_n)$ tuji si števili in je ulomek $\dfrac{B}{A}$ v (3.12) pokrajšan. Zato je

$$A = Q_n(A_1, A_2, \ldots, A_n) \cdot d \quad \text{in} \quad B = Q_{n-1}(A_2, \ldots, A_n) \cdot d.$$

Iz tega sledi $d = \gcd(A, B)$. □

Poglavje 4

Kvocienti v Evklidovem algoritmu

Opisani postopek za računanje verižnih ulomkov deluje tudi za $A < B$. V tem primeru je $A_1 = 0$. Kakšne so frekvence pojavitev delnih kvocientov za takšen B v Evklidovem algoritmu lahko vidimo iz tega, kakšni bodo verižni ulomki za

$$X = \frac{0}{B}, \frac{1}{B}, \frac{2}{B}, \ \ldots, \frac{B-1}{B}.$$

Če predpostavimo, da je B zelo veliko število, lahko preučujemo enostavne verižne ulomke, ko je $X \in_{\mathbb{R}} [0, 1)$. Vpeljemo zaporedje slučajnih spremenljivk $\{X_n\}_{n=1}^{\infty}$, kot smo ga definirali v (3.6):

$$X_0 = X \quad \text{in} \quad X_{n+1} = \frac{1}{X_n} - \left\lfloor \frac{1}{X_n} \right\rfloor. \tag{4.1}$$

Ker je X zvezno porazdeljena slučajna spremenljivka, je za vsak $n \in \mathbb{N}$ takšna tudi slučajna spremenljivka X_n. Definirajmo porazdelitveno funkcijo $F_n(x)$:

$$F_n(x) = P(X_n \leq x), \qquad \text{za } 0 \leq x < 1. \tag{4.2}$$

Z uporabo te porazdelitvene funkcije bomo kasneje izračunali porazdelitev kvocientov v Evklidovem algoritmu, najprej pa si oglejmo nekaj osnovnih lastnosti porazdelitvenih funkcij.

4.1 Porazdelitvene funkcije

Ponovimo osnovne lastnosti porazdelitvenih funkcij.

4.1.1 Diskretne slučajne spremenljivke

Verjetnostna porazdelitev diskretne slučajne spremenljivke je funkcija, ki vrača verjetnost, da ima diskretna slučajna spremenljivka točno določeno vrednost. Označujemo jo s $p(x)$, zanjo velja, da je za vsako število x:

$$p(x) = P(X = x).$$

Pri tem velja še:

1. $p(x) \geq 0$, za vsak x,
2. $\sum_x p(x) = 1$.

Porazdelitvena funkcija opisuje verjetnostno porazdelitev slučajne spremenljivke X. Označujemo jo z $F(x)$. Za diskretno slučajno spremenljivko X s funkcijo verjetnosti $p(x)$, je definirana za vsako število x z

$$F(x) = P(X \leq x) = \sum_{\substack{y \in \mathbb{R} \\ y \leq x}} p(y).$$

Zanjo velja še:

1. $\lim_{x \to -\infty} F(x) = 0$, $\lim_{x \to \infty} F(x) = 1$,
2. če sta a in b realni števili, za kateri velja $a < b$, potem je $F(a) \leq F(b)$,
3. označimo z a^- prvo vrednost od X, ki je manjša od a. Potem je

$$P(a \leq X \leq b) = P(X \leq b) - P(X < a) = F(b) - F(a^-).$$

To velja za vse realne a, b, kjer je $a < b$.

4.1.2 Zvezne slučajne spremenljivke

Funkcija gostote verjetnosti zvezne slučajne spremenljivke X je realna funkcija $f(x) \geq 0$, za $-\infty < x < \infty$, za katero velja:

$$P(a \leq X \leq b) = \int_a^b f(x)\,\mathrm{d}x, \quad \text{za } a, b \in \mathbb{R}, a \leq b.$$

Funkcija gostote verjetnosti vrača relativno verjetnost, da bo zvezna slučajna spremenljivka imela točno določeno vrednost iz množice možnih vrednosti. Zanjo velja še:

1. $\int_{-\infty}^{\infty} f(x)\,\mathrm{d}x = 1,$

2. $P(X = c) = 0$ za vsak $c \in \mathbb{R}$.

Porazdelitvena funkcija $F(x)$ zvezne slučajne spremenljivke X je definirana kot

$$F(x) = P(X \leq x) = \int_{-\infty}^{x} f(y)\,\mathrm{d}y, \quad \text{za } -\infty < x < \infty.$$

Zanjo velja še:

1. $\lim_{x\to-\infty} F(x) = 0,\ \lim_{x\to\infty} F(x) = 1,$

2. če sta a in b realni števili, za kateri velja, da je $a < b$, potem je $F(a) \leq F(b)$,

3. če sta a in b realni števili, za kateri velja, da je $a < b$, velja tudi

$$\begin{aligned} P(a \leq X \leq b) &= \int_a^b f(x)\,\mathrm{d}x = \int_{-\infty}^b f(x)\,\mathrm{d}x - \int_{-\infty}^a f(x)\,\mathrm{d}x \\ &= P(X \leq b) - P(X < a) = F(b) - F(a), \end{aligned}$$

4. funkcijo $f(x)$ lahko dobimo iz $F(x)$ z odvajanjem: $f(x) = F'(x)$.

4.2 Porazdelitvena funkcija $F_n(x)$

Sedaj, ko smo ponovili osnovne lastnosti porazdelitvenih funkcij, lahko izračunamo porazdelitveno funkcijo $F_n(x)$.

Izrek 4.1. *Za porazdelitveno funkcijo $F_n(x)$ velja*

$$\begin{aligned} F_0(x) &= x, \\ F_{n+1}(x) &= \sum_{k\geq 1} \left(F_n\left(\frac{1}{k}\right) - F_n\left(\frac{1}{k+x}\right)\right). \end{aligned} \tag{4.3}$$

Dokaz. Iz definicij (4.1) in (4.2) ter dejstva, da je X enakomerno porazdeljena, dobimo

$$F_0(x) = P(X_0 \leq x) = P(X \leq x) = x.$$

Spomnimo se rekurzivne zveze (4.1): $X_0 = X$ in $X_{n+1} = \frac{1}{X_n} - \left\lfloor \frac{1}{X_n} \right\rfloor$. Od tod in iz definicije (4.2) sledi:

$$\begin{aligned} F_{n+1}(x) &= P(X_{n+1} \le x) = P\left(\frac{1}{X_n} - A_{n+1} \le x\right) = P\left(\frac{1}{X_n} \le x + A_{n+1}\right) \\ &= P\left(\frac{1}{X_n} \in \bigcup_{k\ge 1}[k, k+x]\right) = P\left(X_n \in \bigcup_{k\ge 1}\left[\frac{1}{k+x}, \frac{1}{k}\right]\right) \\ &= \sum_{k\ge 1} P\left(\frac{1}{k+x} \le X_n \le \frac{1}{k}\right) = \sum_{k\ge 1}\left(F_n\left(\frac{1}{k}\right) - F_n\left(\frac{1}{k+x}\right)\right). \end{aligned}$$

□

Če se $F_n(x)$ približuje limitni porazdelitvi $F_\infty(x) = F(x)$, dobimo:

$$F(x) = \sum_{k\ge 1}\left(F\left(\frac{1}{k}\right) - F\left(\frac{1}{k+x}\right)\right). \tag{4.4}$$

Trditev 4.2. *Funkcija $F(x) = \log_b(1+x)$, za $b > 1$, zadošča relaciji (4.4).*

Dokaz. Oglejmo si delno vsoto vrste v (4.4):

$$\begin{aligned} \sum_{1\le k\le N}\left(F\left(\frac{1}{k}\right) - F\left(\frac{1}{k+x}\right)\right) &= \sum_{1\le k\le N}\left(\log_b\left(1+\frac{1}{k}\right) - \log_b\left(1+\frac{1}{k+x}\right)\right) \\ &= \sum_{1\le k\le N}\left(\log_b\left(\frac{1+k}{k}\right) - \log_b\left(\frac{1+k+x}{k+x}\right)\right) \\ &= \sum_{1\le k\le N}\Big(\log_b(1+k) - \log_b(k) - \log_b(1+k+x) + \log_b(k+x)\Big) \\ &= \log_b(2) - \log_b(1) - \log_b(2+x) + \log_b(1+x) \\ &\quad + \log_b(3) - \log_b(2) - \log_b(3+x) + \log_b(2+x) \\ &\quad + \log_b(4) - \log_b(3) - \log_b(4+x) + \log_b(3+x) + \dots \\ &\quad + \log_b(N) - \log_b(N-1) - \log_b(N+x) + \log_b(N-1+x) \\ &\quad + \log_b(1+N) - \log_b(N) - \log_b(1+N+x) + \log_b(N+x) \\ &= \log_b(1+x) + \log_b(1+N) - \log_b(1+N+x) \\ &= \log_b(1+x) + \log_b\left(\frac{1+N}{1+N+x}\right) = F(x) + \log_b\left(\frac{1+N}{1+N+x}\right). \end{aligned}$$

Ko gre $N \longrightarrow \infty$, gre $\log_b\left(\frac{1+N}{1+N+x}\right) \longrightarrow 0$, zato je

$$\sum_{k\geq 1}\left(F\left(\frac{1}{k}\right) - F\left(\frac{1}{k+x}\right)\right) = F(x).$$

□

Ker za porazdelitveno funkcijo velja $F(1) = 1$, bo ustrezna osnova $b = 2$. Iz tega dobimo funkcijo $F(x) = \log_2(1+x) = \lg(1+x)$. Radi pa bi dobili tudi oceno porazdelitvenih funkcij $F_n(x)$.

Omenjene porazdelitvene funkcije je prvi študiral F. K. Gauss leta 1800, a problema asimpotičnega obnašanja funkcije $F_n(x)$ ni znal rešiti. Leta 1928 je R. O. Kuz'min pokazal, da je $F_n(x) = \lg(1+x) + \mathcal{O}(e^{-A\cdot\sqrt{n}})$ za neko pozitivno konstanto A. Leta 1929 je Paul Lévy napako izboljšal in pokazal, da je $F_n(x) = \lg(1+x) + \mathcal{O}(e^{-A\cdot n})$ za neko pozitivno konstanto A. Problem, kako določiti asimptotično obnašanje $F_n(x) - \lg(1+x)$, je rešil šele Eduard Wirsling leta 1974 in v naslednjem poglavju si bomo ogledali, kako.

4.3 Wirsingova metoda

Naj bo $G : [0,1] \longrightarrow \mathbb{R}$ odvedljiva funkcija z omejenim odvodom. Torej obstaja taka konstanta $M \geq 0$, da je

$$\left|G'(x)\right| \leq M, \qquad \text{za vse } x \in [0,1].$$

Potem z znanim Lagrangeovim izrekom dokažemo neenakost

$$\left|G(x) - G(y)\right| \leq M \cdot |x-y|,$$

ki velja za poljubni števili $x, y \in [0,1]$. Z njeno uporabo in z uporabo Weierstrassovega izreka (glej [13], 7. poglavje, Izrek 7.10) dokažemo absolutno in enakomerno konvergenco vrste

$$\sum_{k\geq 1}\left(G\left(\frac{1}{k}\right) - G\left(\frac{1}{k+x}\right)\right), \quad \text{za } x \in [0,1], \tag{4.5}$$

saj velja ocena

$$\begin{aligned}\sum_{k\geq 1}\left|G\left(\frac{1}{k}\right) - G\left(\frac{1}{k+x}\right)\right| &\leq M \cdot \sum_{k\geq 1}\left|\frac{1}{k} - \frac{1}{k+x}\right| \\ &\leq M \cdot \sum_{k\geq 1}\left|\frac{1}{k} - \frac{1}{k+1}\right| = M.\end{aligned}$$

Torej lahko definiramo funkcijo $SG : [0,1] \longrightarrow \mathbb{R}$ z vrsto

$$\big(SG\big)(x) = \sum_{k\geq 1}\left(G\Big(\frac{1}{k}\Big) - G\Big(\frac{1}{k+x}\Big)\right). \tag{4.6}$$

Tako smo definirali operator S, ki funkciji G priredi funkcijo SG.

Trditev 4.3. *Transformacija S je linearna, tj. zanjo veljata aditivnost: $S(G_1 + G_2) = SG_1 + SG_2$, ter homogenost: $S(cG) = c(SG)$, kjer so G_1, G_2 ter G odvedljive funkcije z omejenimi odvodi in je $c \in \mathbb{R}$.*

Dokaz. Aditivnost operatorja S sledi iz

$$\begin{aligned}
\big(S(G_1+G_2)\big)(x) &= \sum_{k\geq 1}\left(\Big(G_1+G_2\Big)\Big(\frac{1}{k}\Big) - \Big(G_1+G_2\Big)\Big(\frac{1}{k+x}\Big)\right)\\
&= \sum_{k\geq 1}\left(G_1\Big(\frac{1}{k}\Big) + G_2\Big(\frac{1}{k}\Big) - G_1\Big(\frac{1}{k+x}\Big) - G_2\Big(\frac{1}{k+x}\Big)\right)\\
&= \sum_{k\geq 1}\left(G_1\Big(\frac{1}{k}\Big) - G_1\Big(\frac{1}{k+x}\Big) + G_2\Big(\frac{1}{k}\Big) - G_2\Big(\frac{1}{k+x}\Big)\right)\\
&= \sum_{k\geq 1}\left(G_1\Big(\frac{1}{k}\Big) - G_1\Big(\frac{1}{k+x}\Big)\right) + \sum_{k\geq 1}\left(G_2\Big(\frac{1}{k}\Big) - G_2\Big(\frac{1}{k+x}\Big)\right)\\
&= \big(SG_1\big)(x) + \big(SG_2\big)(x),
\end{aligned}$$

kjer smo smeli zamenjati vrstni red členov zaradi absolutne konvergence vrst. Ker je

$$\begin{aligned}
\big(S(cG)\big)(x) &= \sum_{k\geq 1}\left(cG\Big(\frac{1}{k}\Big) - cG\Big(\frac{1}{k+x}\Big)\right)\\
&= \sum_{k\geq 1}\left(c\Big(G\Big(\frac{1}{k}\Big) - G\Big(\frac{1}{k+x}\Big)\Big)\right)\\
&= c\cdot\sum_{k\geq 1}\left(G\Big(\frac{1}{k}\Big) - G\Big(\frac{1}{k+x}\Big)\right)\\
&= c\big(SG\big)(x),
\end{aligned}$$

je S tudi homogen operator. □

Če vrsto v (4.5) členoma odvajamo, dobimo vrsto

$$\sum_{k\geq 1}\frac{1}{(k+x)^2}\cdot G'\Big(\frac{1}{k+x}\Big),$$

ki tudi enakomerno in absolutno konvergira, saj velja

$$\begin{aligned}\sum_{k\geq 1}\frac{1}{(k+x)^2}\cdot\left|G'\left(\frac{1}{k+x}\right)\right| &\leq M\cdot\sum_{k\geq 1}\frac{1}{(k+x)^2}\\ &\leq M\cdot\sum_{k\geq 1}\frac{1}{k^2}<\infty.\end{aligned}$$

Po znanem izreku (glej [16], XV. poglavje (Teorija vrst), 6. razdelek) je potem funkcija SG odvedljiva in velja

$$(SG)'(x) = \sum_{k\geq 1}\frac{1}{(k+x)^2}\cdot G'\left(\frac{1}{k+x}\right).$$

S tem smo dokazali tudi, da ima funkcija SG omejen odvod.

Z uporabo funkcije SG in še nekaterih funkcij in operatorjev, ki jih bomo sproti definirali, bomo sedaj določili asimptotično obnašanje $F_n(x)-\lg(1+x)$. Definirajmo za začetek funkcije

$$\begin{aligned}H &= SG,\\ g(x) &= (1+x)\cdot G'(x),\\ h(x) &= (1+x)\cdot H'(x).\end{aligned}$$

Dobimo

$$\begin{aligned}G'\left(\frac{1}{k+x}\right) &= \frac{g\left(\frac{1}{k+x}\right)}{\left(1+\frac{1}{k+x}\right)}\\ &= \left(\frac{k+x}{k+x+1}\right)\cdot g\left(\frac{1}{k+x}\right).\end{aligned}$$

Vstavimo rezultat v definicijo funkcije $h(x)$ in dobimo

$$\begin{aligned}h(x) &= (1+x)\cdot H'(x)\\ &= \sum_{k\geq 1}(1+x)\cdot\left(\frac{1}{k+x}\right)^2\cdot G'\left(\frac{1}{k+x}\right)\\ &= \sum_{k\geq 1}\frac{1+x}{k+x}\cdot\frac{1}{k+x+1}\cdot g\left(\frac{1}{k+x}\right)\\ &= \sum_{k\geq 1}\left(\frac{k}{k+x+1}-\frac{k-1}{k+x}\right)\cdot g\left(\frac{1}{k+x}\right).\end{aligned}\tag{4.7}$$

Vpeljimo novo linearno transformacijo T, tako da velja:

$$h = Tg.$$

Če je funkcija g zvezno odvedljiva, lahko Tg členoma odvajamo in dobimo

$$\begin{aligned}
(Tg)'(x) &= \sum_{k\geq 1}\Bigg(\Bigg(k\cdot(-1)\cdot\Big(\frac{1}{k+x+1}\Big)^2-(k-1)\cdot(-1)\cdot\Big(\frac{1}{k+x}\Big)^2\Bigg)\cdot g\Big(\frac{1}{k+x}\Big)\\
&\quad+\Big(\frac{k}{k+x+1}-\frac{k-1}{k+x}\Big)\cdot\Big((-1)\cdot\Big(\frac{1}{k+x}\Big)^2\cdot g'\Big(\frac{1}{k+x}\Big)\Big)\Bigg)\\
&= \sum_{k\geq 1}\Bigg(g\Big(\frac{1}{k+x}\Big)\cdot\Big(\frac{k-1}{(k+x)^2}-\frac{k}{(k+x+1)^2}\Big)\\
&\quad-g'\Big(\frac{1}{k+x}\Big)\cdot\Big(\frac{1}{k+x}\Big)^2\cdot\Big(\frac{k}{k+x+1}-\frac{k-1}{k+x}\Big)\Bigg)\\
&= \sum_{k\geq 2}g\Big(\frac{1}{k+x}\Big)\cdot\frac{k-1}{(k+x)^2}\\
&\quad-\sum_{k\geq 1}g\Big(\frac{1}{k+x}\Big)\cdot\frac{k}{(k+x+1)^2}\\
&\quad-\sum_{k\geq 1}g'\Big(\frac{1}{k+x}\Big)\cdot\Big(\frac{1}{k+x}\Big)^2\cdot\Big(\frac{k}{k+x+1}-\frac{k-1}{k+x}\Big).
\end{aligned}$$

V prvo vsoto vpeljemo $m = k-1$ in dobimo

$$\begin{aligned}
(Tg)'(x) &= \sum_{m\geq 1}g\Big(\frac{1}{m+1+x}\Big)\cdot\frac{m}{(m+1+x)^2}\\
&\quad-\sum_{k\geq 1}g\Big(\frac{1}{k+x}\Big)\cdot\frac{k}{(k+x+1)^2}\\
&\quad-\sum_{k\geq 1}g'\Big(\frac{1}{k+x}\Big)\cdot\Big(\frac{1}{k+x}\Big)^2\cdot\Big(\frac{k}{k+x+1}-\frac{k-1}{k+x}\Big)\\
&= -\sum_{k\geq 1}\Bigg(\Big(g\Big(\frac{1}{k+x}\Big)-g\Big(\frac{1}{k+1+x}\Big)\Big)\cdot\frac{k}{(k+1+x)^2}\\
&\quad+g'\Big(\frac{1}{k+x}\Big)\cdot\frac{1+x}{(k+x)^3\cdot(k+x+1)}\Bigg).
\end{aligned}$$

Za lažji zapis v nadaljevanju definirajmo funkcijo $\rho(x) = g'(x)$, $\rho(x) : [0,1] \longrightarrow \mathbb{R}_0^+$ in vpeljimo še zadnjo transformacijo U, za katero velja

$$(Tg)' = -U(g'). \tag{4.8}$$

Tako lahko zgornjo vsoto zapišemo kot

$$\begin{aligned}\big(U\rho\big)(x) &= \sum_{k\geq 1}\Bigg(\frac{k}{(k+1+x)^2}\cdot\int_{1/(k+1+x)}^{1/(k+x)}\rho(t)\,\mathrm{d}t \\ &\quad + \frac{1+x}{(k+x)^3\cdot(k+x+1)}\cdot\rho\Big(\frac{1}{k+x}\Big)\Bigg).\end{aligned}$$

Oglejmo si, kakšna je povezava med našim problemom in vsem do sedaj opisanim. Za operator S po enačbi (4.3) velja:

$$F_{n+1}(x) = (SF_n)(x) = \big(S(SF_{n-1})\big)(x) = (S^2F_{n-1})(x) = \cdots,$$

torej

$$F_n(x) = (S^nF_0)(x).$$

Če vzamemo

$$\begin{aligned}F_n(x) &= \lg(1+x) + R_n\big(\lg(1+x)\big), \\ f_n(x) &= (1+x)\cdot F_n'(x),\end{aligned} \tag{4.9}$$

dobimo

$$\begin{aligned}f_n(x) &= (1+x)\Big(\lg(1+x) + R_n\big(\lg(1+x)\big)\Big)' \\ &= (1+x)\Big(\frac{1}{(\ln 2)(1+x)} + \frac{1}{(\ln 2)(1+x)}\cdot R_n'\big(\lg(1+x)\big)\Big) \\ &= \frac{1}{\ln 2}\cdot\Big(1 + R_n'\big(\lg(1+x)\big)\Big).\end{aligned}$$

Odvod te funkcije je enak

$$\begin{aligned}f_n'(x) &= \left(\frac{1}{\ln 2}\Big(1 + R_n'\Big(\lg(1+x)\Big)\Big)\right)' \\ &= \left(\frac{1}{\ln 2} + \frac{1}{\ln 2}\cdot R_n'\Big(\lg(1+x)\Big)\right)' \\ &= \frac{1}{\ln 2}\cdot\left(R_n'\Big(\lg(1+x)\Big)\right)' \\ &= \frac{1}{\ln 2}\cdot\frac{1}{(\ln 2)(1+x)}\cdot R_n''\Big(\lg(1+x)\Big) \\ &= \frac{1}{(\ln 2)^2(1+x)}\cdot R_n''\Big(\lg(1+x)\Big).\end{aligned} \tag{4.10}$$

Trditev 4.4. *Za funkcijo f_n velja*

$$f_n = T^nf_0.$$

Dokaz. Za računanje funkcije $\big(Tf_n\big)(x)$ uporabimo zvezo (4.7), za računanje $f_{n+1}(x)$ pa (4.3). Najprej pokažimo, da velja $f_1 = Tf_0$. V ta namen izračunajmo prvih nekaj vrednosti:

$$\begin{aligned}
F_0(x) &= x, \qquad F_0'(x) = 1,\\
F_1(x) &= \sum_{k\geq 1}\left(F_0\Big(\frac{1}{k}\Big) - F_0\Big(\frac{1}{k+x}\Big)\right) = \sum_{k\geq 1}\left(\frac{1}{k} - \frac{1}{k+x}\right),\\
F_1'(x) &= \sum_{k\geq 1}\frac{1}{(k+x)^2},\\
f_0(x) &= (1+x)F_0'(x) = 1+x,\\
f_1(x) &= (1+x)F_1'(x) = (1+x)\sum_{k\geq 1}\frac{1}{(k+x)^2}.
\end{aligned}$$

Izračunajmo še funkcijo $\big(Tf_0\big)(x)$:

$$\begin{aligned}
\big(Tf_0\big)(x) &= \sum_{k\geq 1}\left(\frac{k}{k+x+1} - \frac{k-1}{k+x}\right)\cdot f_0\Big(\frac{1}{k+x}\Big)\\
&= \sum_{k\geq 1}\left(\frac{k}{k+x+1} - \frac{k-1}{k+x}\right)\cdot\left(1+\frac{1}{k+x}\right)\\
&= \sum_{k\geq 1}\left(\frac{k(k+x)-(k-1)(k+x+1)}{(k+x+1)(k+x)}\right)\cdot\left(1+\frac{1}{k+x}\right)\\
&= \sum_{k\geq 1}\left(\frac{k^2+kx-k^2-kx-k+k+x+1}{(k+x+1)(k+x)}\right)\cdot\left(\frac{k+x+1}{k+x}\right)\\
&= \sum_{k\geq 1}\frac{x+1}{(k+x)^2}.
\end{aligned}$$

Pokažimo sedaj, da velja $f_{n+1} = Tf_n$:

$$\begin{aligned}
\big(Tf_n\big)(x) &= \sum_{k\geq 1}\left(\frac{k}{k+x+1} - \frac{k-1}{k+x}\right)\cdot f\Big(\frac{1}{k+x}\Big)\\
&= \sum_{k\geq 1}\left(\frac{k(k+x)-(k-1)(k+x+1)}{(k+x+1)(k+x)}\right)\cdot F_n'\Big(\frac{1}{k+x}\Big)\left(1+\frac{1}{k+x}\right)\\
&= \sum_{k\geq 1}\frac{1+x}{(k+x)^2}\cdot F_n'\Big(\frac{1}{k+x}\Big).
\end{aligned}$$

$$\begin{aligned} f_{n+1}(x) &= (1+x)F'_{n+1}(x) \\ &= (1+x)\sum_{k\geq 1}\left(F'_n\left(\frac{1}{k}\right) - F'_n\left(\frac{1}{k+x}\right)\right) \\ &= (1+x)\sum_{k\geq 1}\frac{1}{(k+x)^2}\cdot F'_n\left(\frac{1}{k+x}\right) \\ &= \sum_{k\geq 1}\frac{1+x}{(k+x)^2}\cdot F'_n\left(\frac{1}{k+x}\right). \end{aligned}$$

Rekurzivno dobimo

$$f_{n+1} = \big(Tf_n\big)(x) = \big(T(Tf_{n-1})\big)(x) = \big(T^2 f_{n-1}\big)(x) = \cdots = \big(T^{n+1}f_0\big)(x).$$

□

Vstavimo funkcijo f_n v zvezo $(Tg)' = -U(g')$, ki smo jo definirali v (4.8). Dobimo

$$f'_n = (T^n f_0)' = (-1)^n U^n f'_0.$$

Funkciji F_n in f_n sta n–krat zvezno odvedljivi. Zato iz (4.10) sledi

$$f'_n(x) = \frac{1}{(\ln 2)^2\cdot(1+x)}\cdot R''_n\Big(\lg(1+x)\Big) = (-1)^n U^n f'_0.$$

Če enačbo preuredimo, dobimo

$$(-1)^n\cdot R''_n\Big(\lg(1+x)\Big) = (\ln 2)^2\cdot(1+x)\cdot U^n f'_0. \tag{4.11}$$

Začetni funkciji sta:

$$F_0(x) = x \quad \text{in} \quad f_0(x) = (1+x)\cdot F'_0(x) = 1+x.$$

Opišimo na tem mestu na kratko nekaj lastnosti deljenih diferenc, ki jih bomo potrebovali za določanje ostanka $R_n(x)$.
Newtonov interpolacijski polinom $P_n(x)$ je polinom, ki se v točkah $x_0,\ldots,x_n$ ujema s funkcijo f. Zapišemo ga kot

$$\begin{aligned} P_n(x) &= f[x_0] + f[x_0,x_1](x-x_0) + f[x_0,x_1,x_2](x-x_0)(x-x_1) \\ &\quad + \cdots + f[x_0,x_1,\ldots,x_n](x-x_0)\cdots(x-x_{n-1}). \end{aligned}$$

Pri tem je $f[x_0,x_1,\ldots,x_k]$ *deljena diferenca*, tj. vodilni koeficient (pri x^k) interpolacijskega polinoma stopnje k, ki se ujema z f v paroma različnih točkah $x_0,x_1,\ldots,x_k$.

Za deljene diference velja naslednja rekurzivna formula:

$$\begin{aligned} f[x_0] &= f(x_0), \\ f[x_0, x_1, \ldots, x_n] &= \frac{f[x_1, \ldots, x_n] - f[x_0, \ldots, x_{n-1}]}{x_n - x_0}. \end{aligned} \tag{4.12}$$

Oglejmo si deljeni diferenci za $k = 1$ in $k = 2$:

$$\begin{aligned} f[x_0, x_1] &= \frac{f[x_1] - f[x_0]}{x_1 - x_0} \\ &= \frac{f(x_1) - f(x_0)}{x_1 - x_0} \\ &= \frac{f(x_0)}{x_0 - x_1} + \frac{f(x_1)}{x_1 - x_0}, \end{aligned}$$

$$\begin{aligned} f[x_0, x_1, x_2] &= \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0} \\ &= \frac{\frac{f(x_2)-f(x_1)}{x_2-x_1} - \frac{f(x_1)-f(x_0)}{x_1-x_0}}{x_2 - x_0} \\ &= \frac{\big(f(x_2) - f(x_1)\big)(x_1 - x_0) - \big(f(x_1) - f(x_0)\big)(x_2 - x_1)}{(x_2 - x_0)(x_2 - x_1)(x_1 - x_0)} \\ &= \frac{f(x_0)(x_2 - x_1) + f(x_1)(-x_1 + x_0 - x_2 + x_1) + f(x_2)(x_1 - x_0)}{(x_2 - x_0)(x_2 - x_1)(x_1 - x_0)} \\ &= \frac{f(x_0)}{(x_0 - x_1)(x_0 - x_2)} + \frac{f(x_1)}{(x_1 - x_0)(x_1 - x_2)} + \frac{f(x_2)}{(x_2 - x_0)(x_2 - x_1)}. \end{aligned}$$

Izrek 4.5. *Iz definicije* (4.12) *lahko izpeljemo drugačen zapis deljene diference:*

$$f[x_0, x_1, \ldots, x_n] = \sum_{0 \le k \le n} \frac{f(x_k)}{\prod\limits_{\substack{0 \le j \le n \\ j \ne k}} (x_k - x_j)}.$$

Dokaz. Za dokazovanje uporabimo indukcijo. Da to velja za $f[x_0, x_1]$ in za $f[x_0, x_1, x_2]$, lahko vidimo že iz zgornjih primerov. Predpostavimo, da to velja za $f[x_0, x_1, \ldots, x_n]$ in

dokažimo, da velja tudi za $f[x_0, x_1, \ldots, x_{n+1}]$.

$$\begin{aligned}
f[x_0, x_1, \ldots, x_{n+1}] &= \frac{f[x_1, \ldots, x_{n+1}] - f[x_0, \ldots, x_n]}{x_{n+1} - x_0} \\
&= \left(\sum_{1 \le k \le n+1} \frac{f(x_k)}{\prod\limits_{\substack{1 \le j \le n+1 \\ j \ne k}} (x_k - x_j)} - \sum_{0 \le k \le n} \frac{f(x_k)}{\prod\limits_{\substack{0 \le j \le n \\ j \ne k}} (x_k - x_j)} \right) \Big/ \Big(x_{n+1} - x_0 \Big) \\
&= \left(\sum_{0 \le k \le n+1} \frac{f(x_k)(x_k - x_0)}{\prod\limits_{\substack{0 \le j \le n+1 \\ j \ne k}} (x_k - x_j)} - \sum_{0 \le k \le n+1} \frac{f(x_k)(x_k - x_{n+1})}{\prod\limits_{\substack{0 \le j \le n+1 \\ j \ne k}} (x_k - x_j)} \right) \Big/ \Big(x_{n+1} - x_0 \Big) \\
&= \left(\sum_{0 \le k \le n+1} \frac{f(x_k)(x_k - x_0 - x_k + x_{n+1})}{\prod\limits_{\substack{0 \le j \le n+1 \\ j \ne k}} (x_k - x_j)} \right) \Big/ \Big(x_{n+1} - x_0 \Big) \\
&= \sum_{0 \le k \le n+1} \frac{f(x_k)}{\prod\limits_{\substack{0 \le j \le n+1 \\ j \ne k}} (x_k - x_j)}.
\end{aligned}$$

□

Za k–krat zvezno odvedljivo funkcijo f velja

$$f[x_0, x_1, \ldots, x_k] = \frac{1}{k!} f^{(k)}(\xi), \qquad \text{pri čemer je } \min_{i=0,\ldots,k}(x_i) < \xi < \max_{i=0,\ldots,k}(x_i). \tag{4.13}$$

Sedaj, ko smo si ogledali osnovne lastnosti deljenih diferenc, izpeljimo formulo za določanje ostanka $R_n(x)$.

Izrek 4.6. *Za ostanek $R_n(x)$ velja*

$$R_n(x) = \frac{x(1-x)}{2} \cdot R_n''\big(\xi(x)\big), \tag{4.14}$$

za neko funkcijo $\xi(x)$, za katero velja $0 \le \xi(x) \le 1$, ko je $0 \le x \le 1$.

Dokaz. Za dokazovanje bomo uporabili deljene diference. Najprej pokažimo, da velja $R_n(0) = R_n(1) = 0$. Funkcijo $F_n(x)$ smo definirali kot $F_n(x) = P(X_n \le x)$. Vemo, da je $F_n(x) = \lg(1+x) + R_n(\lg(1+x))$. Če vstavimo $x = 0$, dobimo:

$$P(X_n \le 0) = 0 = F_n(0) = \lg(1) + R_n(\lg(1)) = 0 + R_n(0), \text{ torej } R_n(0) = F_n(0) = 0.$$

Če pa vstavimo $x = 1$, dobimo:

$$P(X_n \le 1) = 1 = F_n(1) = \lg(2) + R_n(\lg(2)) = 1 + R_n(1), \text{ torej } R_n(1) = F_n(1) - 1 = 0.$$

Izberimo sedaj tri različne točke: $x_0 = 0$, $x_1 = x$, $x_2 = 1$, kjer $0 < x < 1$. Zaradi definicije (4.13) velja

$$R_n[x_0, x_1, x_2] = \frac{1}{2!} \cdot R_n''\big(\xi(x)\big),$$

pri čemer je $0 < \xi(x) < 1$.

Po drugi strani pa je deljena diferenca ostanka na zgornjih treh točkah enaka

$$R_n[0, x, 1] = \frac{R_n(0)}{(0-x)(0-1)} + \frac{R_n(x)}{(x-0)(x-1)} + \frac{R_n(1)}{(1-0)(1-x)} = \frac{R_n(x)}{x(x-1)}.$$

Iz tega sledi, da je

$$\frac{R_n(x)}{x(x-1)} = \frac{1}{2!} \cdot R_n''\big(\xi(x)\big) \quad \text{oziroma} \quad R_n(x) = -\frac{x(1-x)}{2} \cdot R_n''\big(\xi(x)\big).$$

□

V nadaljevanju bomo pokazali, da transformacija U^n konstantno funkcijo spremeni v funkcijo z zelo majhnimi vrednostmi. Posledično to pomeni, da je zaradi (4.11) tudi $|R_n''(x)|$ zelo majhen za $0 \le x < 1$. Izrek 4.3 pa nam zagotovi, da je majhen prav tako $R_n(x)$.

Linearna transformacija U je pozitivna, kar pomeni, da je $\big(U\rho\big)(x) \ge 0$ za vsak x, če je $\rho(x) \ge 0$ za vsak x. Je tudi monotona: če je $\rho_1(x) \le \rho_2(x)$ za vsak x, potem je $\big(U\rho_1\big)(x) \le \big(U\rho_2\big)(x)$ za vsak x. Zato lahko poiščemo takšno funkcijo ρ, za katero bomo lahko natančno izračunali $U\rho$. S tem pa bomo prišli do omejitev, ki nas zanimajo.

Najprej poiščimo funkcijo g, tako da je Tg enostavno izračunati. Če vzamemo, da so funkcije definirane za vse $x \ge 0$, namesto le za $0 \le x < 1$, dobimo iz enačbe (4.6):

$$\begin{aligned}
\big(SG\big)(x+1) - \big(SG\big)(x) &= \sum_{k\ge1}\left(G\left(\frac{1}{k}\right) - G\left(\frac{1}{k+x+1}\right)\right) \\
&\quad -\sum_{k\ge1}\left(G\left(\frac{1}{k}\right) - G\left(\frac{1}{k+x}\right)\right) \\
&= -G\left(\frac{1}{x+2}\right) - G\left(\frac{1}{x+3}\right) - \cdots \\
&\quad + G\left(\frac{1}{x+1}\right) + G\left(\frac{1}{x+2}\right) + \cdots \\
&= G\left(\frac{1}{x+1}\right) - \lim_{k\to\infty} G\left(\frac{1}{k+x}\right) \\
&= G\left(\frac{1}{x+1}\right) - G(0),
\end{aligned}$$

kadar je G zvezna funkcija. Iz definicij funkcij in transformacij sledi

$$\big(Tg\big)(x) = h(x) = (1+x)\cdot H'(x) = (1+x)\cdot (SG)'(x).$$

Če v to zvezo vstavimo definicijo $g(x) = (1+x)\cdot G'(x)$, dobimo

$$T((1+x)\cdot G') = (1+x)\cdot (SG)'.$$

Iz tega sledi

$$\begin{aligned}
\frac{\big(Tg\big)(x)}{x+1} - \frac{\big(Tg\big)(x+1)}{x+2} &= \frac{(1+x)\cdot (SG)'(x)}{x+1} - \frac{(2+x)\cdot (SG)'(x+1)}{x+2} \\
&= \big(SG\big)'(x) - \big(SG\big)'(x+1) \\
&= \sum_{k\geq 1}\left(\frac{1}{k+x}\right)^2 \cdot G'\left(\frac{1}{k+x}\right) \\
&\quad -\sum_{k\geq 1}\left(\frac{1}{k+x+1}\right)^2 \cdot G'\left(\frac{1}{k+x+1}\right) \\
&= \left(\frac{1}{1+x}\right)^2 \cdot G'\left(\frac{1}{1+x}\right) + \left(\frac{1}{2+x}\right)^2 \cdot G'\left(\frac{1}{2+x}\right) + \cdots \\
&\quad -\left(\frac{1}{2+x}\right)^2 \cdot G'\left(\frac{1}{2+x}\right) - \left(\frac{1}{3+x}\right)^2 \cdot G'\left(\frac{1}{3+x}\right) - \cdots \\
&= \left(\frac{1}{1+x}\right)^2 \cdot G'\left(\frac{1}{1+x}\right) - \lim_{k\to\infty}\left(\frac{1}{k+x}\right)^2 \cdot G'\left(\frac{1}{k+x}\right) \\
&= \left(\frac{1}{1+x}\right)^2 \cdot G'\left(\frac{1}{1+x}\right).
\end{aligned} \tag{4.15}$$

Če v enačbi $g(x) = (1+x)\cdot G'(x)$ izberemo $x = \dfrac{1}{1+y}$, dobimo

$$g\left(\frac{1}{1+x}\right) = \left(1 + \left(\frac{1}{1+x}\right)\right)\cdot G'\left(\frac{1}{1+x}\right),$$

iz česar sledi

$$G'\left(\frac{1}{1+x}\right) = g\left(\frac{1}{1+x}\right)\cdot \frac{1+x}{2+x}.$$

Vstavimo to v enačbo (4.15):

$$\begin{aligned}
\frac{\big(Tg\big)(x)}{x+1} - \frac{\big(Tg\big)(x+1)}{x+2} &= \left(\frac{1}{1+x}\right)^2 \cdot g\left(\frac{1}{1+x}\right)\cdot \frac{1+x}{2+x} \\
&= \frac{1}{1+x}\cdot\frac{1}{2+x}\cdot g\left(\frac{1}{1+x}\right).
\end{aligned} \tag{4.16}$$

Ker je $\dfrac{1}{1+x}\cdot\dfrac{1}{2+x}=\dfrac{1}{1+x}-\dfrac{1}{2+x}$, velja

$$\frac{Tg(x)}{x+1}-\frac{Tg(x+1)}{x+2} \;=\; \left(\frac{1}{1+x}-\frac{1}{2+x}\right)\cdot g\left(\frac{1}{1+x}\right).$$

Če vzamemo, da je $Tg(x)=\dfrac{1}{1+x}$, pa dobimo

$$\frac{Tg(x)}{x+1}-\frac{Tg(x+1)}{x+2} \;=\; \frac{\frac{1}{1+x}}{x+1}-\frac{\frac{1}{2+x}}{x+2} \;=\; \frac{2x+3}{(x+1)^2\cdot(x+2)^2}. \tag{4.17}$$

Združimo (4.16) in (4.17):

$$\begin{aligned}\frac{2x+3}{(x+1)^2\cdot(x+2)^2} &= \frac{1}{1+x}\cdot\frac{1}{2+x}\cdot g\left(\frac{1}{1+x}\right),\\ \frac{2x+3}{(x+1)\cdot(x+2)} &= g\left(\frac{1}{1+x}\right).\end{aligned}$$

Izberimo $y=\dfrac{1}{1+x}$, kar pomeni, da je $x=\dfrac{1}{y}-1$. Dobimo

$$\begin{aligned}g(y) &= y\cdot\frac{2(\frac{1}{y}-1)+3}{((\frac{1}{y}-1)+2)}=y\cdot\frac{\frac{2}{y}+1}{\frac{1}{y}+1}=y\cdot\frac{2+y}{1+y}\\ &= \frac{2y+y^2}{1+y}=\frac{(y+1)^2-1}{1+y}=y+1-\frac{1}{1+y}.\end{aligned}$$

Naj bo $\rho(x)=g'(x)=1+\dfrac{1}{(1+x)^2}$. Iz $(Tg)'=-U(g')$ dobimo

$$\big(U\rho\big)(x)=-(Tg(x))'=\frac{1}{(1+x)^2}.$$

To je funkcija ρ, ki smo jo iskali. Za to izbiro funkcije ρ velja

$$\frac{\rho(x)}{\big(U\rho\big)(x)}=\frac{1+\frac{1}{(1+x)^2}}{\frac{1}{(1+x)^2}}=(1+x)^2+1. \tag{4.18}$$

Omejimo se sedaj nazaj na $0\le x\le 1$. Vrednost $\dfrac{\rho(x)}{\big(U\rho\big)(x)}$ lahko po formuli (4.18) omejimo z

$$2\le\frac{\rho(x)}{\big(U\rho\big)(x)}\le 5 \quad \text{oziroma} \quad \frac{\rho}{5}\le U\rho\le\frac{\rho}{2}.$$

Ker sta U in ρ pozitivni, veljajo tudi naslednje neenakosti:

$$\begin{aligned}
\frac{\rho}{5} \le U\rho \;\Rightarrow\; \frac{\rho}{25} \le \frac{U\rho}{5} \quad \text{in} \quad \frac{U\rho}{5} \le U^2\rho \;\Rightarrow\; \frac{\rho}{25} \le \frac{U\rho}{5} \le U^2\rho,\\
U\rho \le \frac{\rho}{2} \;\Rightarrow\; \frac{U\rho}{2} \le \frac{\rho}{4} \quad \text{in} \quad U^2\rho \le \frac{U\rho}{2} \;\Rightarrow\; U^2\rho \le \frac{U\rho}{2} \le \frac{\rho}{4},
\end{aligned}$$

torej

$$\frac{\rho}{25} \le U^2\rho \le \frac{\rho}{4}.$$

Po $n-1$ korakih tako za zgoraj izbrano funkcijo ρ velja

$$5^{-n}\rho \le U^n\rho \le 2^{-n}\rho. \tag{4.19}$$

Prej smo izračunali funkcijo $f_0(x) = 1+x$. Sedaj definirajmo konstantno funkcijo $\chi(x) = f_0'(x) = 1$ in ker je $\rho(x) = 1 + \dfrac{1}{(1+x)^2}$, zanjo velja

$$\frac{5}{4}\chi \le \rho \le 2\,\chi\,, \quad \text{za } 0 \le x \le 1. \tag{4.20}$$

Če združimo neenačbi iz (4.19) in (4.20), dobimo

$$\begin{aligned}
\tfrac{5}{4}\chi \le \rho &\implies \tfrac{5}{8}5^{-n}\chi \le \tfrac{1}{2}5^{-n}\rho,\\
5^{-n}\rho \le U^n\rho &\implies \tfrac{1}{2}5^{-n}\rho \le \tfrac{1}{2}U^n\rho,\\
\tfrac{1}{2}\rho \le \chi &\implies \tfrac{1}{2}U^n\rho \le U^n\chi,\\
\chi \le \tfrac{4}{5}\rho &\implies U^n\chi \le \tfrac{4}{5}U^n\rho,\\
U^n\rho \le 2^{-n}\rho &\implies \tfrac{4}{5}U^n\rho \le \tfrac{4}{5}2^{-n}\rho,\\
\rho \le 2\chi &\implies \tfrac{4}{5}2^{-n}\rho \le \tfrac{8}{5}2^{-n}\chi.
\end{aligned}$$

Pomembni deli teh neenačb so

$$\tfrac{5}{8}5^{-n}\chi \le U^n\chi \le \tfrac{8}{5}2^{-n}\chi.$$

Na levi in desni strani upoštevamo $\chi(x) = 1$, na sredini pa $\chi(x) = f_0'(x)$ ter enačbo pomnožimo z $(1+x)(\ln 2)^2$ in dobimo

$$(1+x)(\ln 2)^2 \cdot \tfrac{5}{8}5^{-n} \le (1+x)(\ln 2)^2 \cdot U^n f_0'(x) \le (1+x)(\ln 2)^2 \cdot \tfrac{8}{5}2^{-n}\,,$$

kar je po enačbi (4.11) enako

$$(1+x)(\ln 2)^2 \cdot \tfrac{5}{8}5^{-n} \le (-1)^n \cdot R_n''\Big(\lg(1+x)\Big) \le (1+x)(\ln 2)^2 \cdot \tfrac{8}{5}2^{-n}\,.$$

Če v tej neenačbi definiramo $y = \lg(1+x)$, torej $1+x = 2^y$ (velja $0 \le y \le 1$ za $0 \le x \le 1$), dobimo

$$2^y \cdot (\ln 2)^2 \cdot \tfrac{5}{8} 5^{-n} \le (-1)^n R_n''(y) \le 2^y \cdot (\ln 2)^2 \cdot \tfrac{8}{5} 2^{-n}\,.$$

Za $y = 0$ velja $2^y = 1$, za $y = 1$ pa $2^y = 2$. Iz tega sledi naslednja neenakost:

$$1 \cdot (\ln 2)^2 \cdot \tfrac{5}{8} 5^{-n} \le (-1)^n R_n''(y) \le 2 \cdot (\ln 2)^2 \cdot \tfrac{8}{5} 2^{-n}\,,$$

torej

$$\tfrac{5}{8}(\ln 2)^2 5^{-n} \le (-1)^n R_n''(x) \le \tfrac{16}{5}(\ln 2)^2 2^{-n}, \quad \text{za } 0 \le x \le 1. \tag{4.21}$$

Z uporabo vsega do sedaj izračunanega lahko izpeljemo naslednji izrek:

Izrek 4.7. *Za porazdelitveno funkcijo $F_n(x)$ velja:*

$$F_n(x) = \lg(1+x) + \mathcal{O}(2^{-n}), \quad \textit{ko } n \to \infty.$$

Dokaz. Spomnimo se enačb (4.9) in (4.14):

$$F_n(x) = \lg(1+x) + R_n\big(\lg(1+x)\big),$$
$$R_n(x) = -\frac{x \cdot (1-x)}{2} \cdot R_n''\big(\xi(x)\big),$$

za neko funkcijo $\xi(x)$, za katero velja $0 \le \xi(x) \le 1$, ko je $0 \le x \le 1$. Ko gre $n \to \infty$, lahko drugi odvod ostanka zaradi (4.21) zapišemo kot $R_n''(x) = \mathcal{O}(2^{-n})$. Vrednost $\lg(1+x)$ je za $0 \le x < 1$ znotraj intervala $[0, 1)$, zato lahko ostanek zapišemo kot

$$R_n\big(\lg(1+x)\big) = -\frac{\lg(1+x) \cdot (1 - \lg(1+x))}{2} \cdot R_n''\Big(\xi\big(\lg(1+x)\big)\Big).$$

Ko gre $n \to \infty$, lahko tako tudi ostanek zapišemo kot $R_n(\lg(1+x)) = \mathcal{O}(2^{-n})$. Res je

$$F_n(x) = \lg(1+x) + \mathcal{O}(2^{-n}), \quad \text{ko } n \to \infty.$$

□

4.4 Porazdelitev delnih kvocientov

Rezultati o verjetnostnih porazdelitvah, ki smo jih dobili do sedaj, veljajo za verižne ulomke, ko je X realno število, enakomerno porazdeljeno na intervalu $[0, 1)$. Toda realno število je racionalno z verjetnostjo 0, saj so skoraj vsa realna števila iracionalna. Zato teh rezultatov ne moremo direktno uporabiti za Evklidov algoritem.

Preden lahko uporabimo izrek 4.7, moramo definirati še nekaj pravil. Uporabimo znanje iz osnov teorije mere.

Lema 4.8. *Naj bodo $I_1, I_2, \ldots, J_1, J_2, \ldots$ paroma disjunktni intervali znotraj intervala $[0,1)$ in naj bo*

$$I = \bigcup_{k\geq 1} I_k, \quad J = \bigcup_{k\geq 1} J_k, \quad K = [0,1]\backslash(I \cup J).$$

Recimo, da ima K mero 0. Definirajmo še množico $P_n = \left\{\frac{0}{n}, \frac{1}{n}, \frac{2}{n}, \ldots, \frac{n-1}{n}\right\}$. Potem je

$$\lim_{n\to\infty} \frac{\|I \cap P_n\|}{n} = \mu(I),$$

kjer je $\mu(I)$ Lebesguova mera in zanjo velja $\mu(I) = \mu(I_1) + \mu(I_2) + \cdots = \sum_{k\geq 1} \text{dolžina}(I_k)$, $\|I \cap P_n\|$ pa označuje število elementov množice $I \cap P_n$.

Dokaz. Naj bo $I_N = \bigcup_{1\leq k\leq N} I_k$ in $J_N = \bigcup_{1\leq k\leq N} J_k$. Za $\varepsilon > 0$ poiščimo tak N, da bo

$$\mu(I_N) + \mu(J_N) \geq 1 - \varepsilon,$$

tako da pokrijemo skoraj celoten interval, in naj bo

$$K_N = K \cup \bigcup_{k>N} I_k \cup \bigcup_{k>N} J_k.$$

Če je I interval katerekoli izmed oblik (a,b), $[a,b)$, $(a,b]$ ali $[a,b]$, potem je $\mu(I) = b - a$. Kolikšno je v tem primeru število elementov množice $I \cap P_n$? Recimo, da je $\frac{k}{n} \leq a \leq \frac{k+1}{n}$ in $\frac{\ell}{n} \leq b \leq \frac{\ell+1}{n}$ za $k+1 < \ell$. Možne situacije so podane v tabeli 4.1 (stran 38). Rezultati tabele 4.1, ki nas zanimajo, so povzeti v tabeli 4.2 (stran 38). Možen pa je še en primer, namreč $\frac{m}{n} < a \leq b < \frac{m+1}{n}$. V tem primeru je $0 \leq n\mu(I) < 1$, $\|I \cap P_n\| = 0$, torej je $n\mu(I) - 1 \leq \|I \cap P_n\|$. To je prvi robni primer. Drugega lahko razberemo iz tabele 4.2: $n\mu(I) + 1 \geq \|I \cap P_n\|$. Tako dobimo

$$n\mu(I) - 1 \leq \|I \cap P_n\| \leq n\mu(I) + 1.$$

Naj bo sedaj še $r_n = \|I_N \cap P_n\|$, $s_n = \|J_N \cap P_n\|$ ter $t_n = \|K_N \cap P_n\|$. Ker unija $I_N \cup J_N \cup K_N$ pokrije celoten interval, je

$$\begin{aligned}
& r_n + s_n + t_n = n, \\
n\mu(I_N) - N &= n\mu(I_1) - 1 + n\mu(I_2) - 1 + \ldots \\
&\leq r_n \leq n\mu(I_1) + 1 + n\mu(I_2) + 1 + \cdots = n\mu(I_N) + N, \\
n\mu(J_N) - N &= n\mu(J_1) - 1 + n\mu(J_2) - 1 + \cdots \\
&\leq s_n \leq n\mu(J_1) + 1 + n\mu(J_2) + 1 + \cdots = n\mu(J_N) + N.
\end{aligned}$$

$a = \frac{k}{n}$	$b = \frac{\ell}{n}$	$n\mu(I) = \ell - k$ $\|I \cap P_n\| = l - k + 1$
	$\frac{\ell}{n} < b < \frac{\ell+1}{n}$	$\ell - k < n\mu(I) < \ell - k + 1$ $\|I \cap P_n\| = \ell - k + 1$
	$b = \frac{\ell+1}{n}$	$n\mu(I) = \ell - k + 1$ $\|I \cap P_n\| = \ell - k + 2$
$\frac{k}{n} < a < \frac{k+1}{n}$	$b = \frac{\ell}{n}$	$\ell - k - 1 < n\mu(I) < \ell - k$ $\|I \cap P_n\| = \ell - k$
	$\frac{\ell}{n} < b < \frac{\ell+1}{n}$	$\ell - k - 1 < n\mu(I) < \ell - k + 1$ $\|I \cap P_n\| = \ell - k$
	$b = \frac{\ell+1}{n}$	$\ell - k < n\mu(I) < \ell - k + 1$ $\|I \cap P_n\| = \ell - k + 1$
$a = \frac{k+1}{n}$	$b = \frac{\ell}{n}$	$n\mu(I) = \ell - k - 1$ $\|I \cap P_n\| = \ell - k$
	$\frac{\ell}{n} < b < \frac{\ell+1}{n}$	$\ell - k - 1 < n\mu(I) < \ell - k$ $\|I \cap P_n\| = \ell - k$
	$b = \frac{\ell+1}{n}$	$n\mu(I) = \ell - k$ $\|I \cap P_n\| = \ell - k + 1$

Tabela 4.1: Število elementov množice $I \cap P_n$.

$a = \frac{k}{n}$	$b = \frac{\ell}{n}$	$n\mu(I) = \|I \cap P_n\| - 1$
	$\frac{\ell}{n} < b < \frac{\ell+1}{n}$	$\|I \cap P_n\| - 1 < n\mu(I) < \|I \cap P_n\|$
	$b = \frac{\ell+1}{n}$	$n\mu(I) = \|I \cap P_n\| - 1$
$\frac{k}{n} < a < \frac{k+1}{n}$	$b = \frac{\ell}{n}$	$\|I \cap P_n\| - 1 < n\mu(I) < \|I \cap P_n\|$
	$\frac{\ell}{n} < b < \frac{\ell+1}{n}$	$\|I \cap P_n\| - 1 < n\mu(I) < \|I \cap P_n\| + 1$
	$b = \frac{\ell+1}{n}$	$\|I \cap P_n\| - 1 < n\mu(I) < \|I \cap P_n\|$
$a = \frac{k+1}{n}$	$b = \frac{\ell}{n}$	$n\mu(I) = \|I \cap P_n\| - 1$
	$\frac{\ell}{n} < b < \frac{\ell+1}{n}$	$\|I \cap P_n\| - 1 < n\mu(I) < \|I \cap P_n\|$
	$b = \frac{\ell+1}{n}$	$n\mu(I) = \|I \cap P_n\| - 1$

Tabela 4.2: Število elementov množice $I \cap P_n$ – povzetek.

Ker velja

$$1 - \varepsilon + \mu(I) \leq (\mu(I_N) + \mu(J_N)) + \mu(I) \leq \mu(I_N) + (\mu(J) + \mu(I)) = \mu(I_N) + 1,$$

je $\mu(I_N) \geq \mu(I) - \varepsilon$. Iz

$$1 - \varepsilon \leq \mu(I_N) + \mu(J_N) \leq \mu(I) + \mu(J_N)$$

sledi tudi $\mu(I) + \varepsilon \geq 1 - \mu(J_N)$. Če vse skupaj združimo, dobimo:

$$\begin{aligned} \mu(I) - \tfrac{N}{n} - \varepsilon &\leq \mu(I_N) - \tfrac{N}{n} \leq \tfrac{r_n}{n} \leq \tfrac{r_n + t_n}{n} \\ &= 1 - \tfrac{s_n}{n} \leq 1 - \mu(J_N) + \tfrac{N}{n} \leq \mu(I) + \tfrac{N}{n} + \varepsilon. \end{aligned}$$

To velja za vse n in za vsak ε, zato $\lim\limits_{n\to\infty} \frac{r_n}{n} = \lim\limits_{n\to\infty} \frac{\|I_N \cap P_n\|}{n} = \lim\limits_{n\to\infty} \frac{\|I \cap P_n\|}{n} = \mu(I)$. □

Sedaj lahko združimo izrek 4.7 in lemo 4.8 in izpeljemo nekaj dejstev v zvezi z Evklidovim algoritmom.

Izrek 4.9. *Naj bosta n in k pozitivni celi števili in naj bo $p_k(a, n)$ verjetnost, da je $(k+1)$–ti kvocient A_{k+1} v Evklidovem algoritmu enak a, ko je $B = n$ in je A izbran naključno. Potem velja*

$$\lim_{n\to\infty} p_k(a, n) = F_k\left(\frac{1}{a}\right) - F_k\left(\frac{1}{a+1}\right),$$

kjer je $F_k(x)$ porazdelitvena funkcija iz (4.2).

Dokaz. Množica I vseh X–ov na intervalu $[0, 1)$, za katere je $A_{k+1} = a$, je unija disjunktnih intervalov. Prav tako je množica J vseh X–ov na intervalu $[0, 1)$, za katere $A_{k+1} \neq a$, unija disjunktnih intervalov. Sedaj uporabimo lemo 4.8, kjer je K množica vseh X–ov, za katere je A_{k+1} nedefiniran.
Za razliko porazdelitvenih funkcij velja:

$$\begin{aligned} F_k\left(\frac{1}{a}\right) &= P\left(X_k \leq \frac{1}{a}\right), \\ F_k\left(\frac{1}{a+1}\right) &= P\left(X_k \leq \frac{1}{a+1}\right), \\ F_k\left(\frac{1}{a}\right) - F_k\left(\frac{1}{a+1}\right) &= P\left(\frac{1}{a+1} < X_k \leq \frac{1}{a}\right). \end{aligned}$$

Ker je $A_{k+1} = \left\lfloor \frac{1}{X_k} \right\rfloor$, velja

$$P\left(\frac{1}{a+1} < X_k \leq \frac{1}{a}\right) = P\left(a \leq \frac{1}{X_k} < a+1\right) = P(A_{k+1} = a).$$

In ker je I množica vseh X–ov, za katere je $A_{k+1} = a$, je $P(A_{k+1} = a) = \mu(I)$. Če vse to združimo, dobimo

$$F_k\left(\frac{1}{a}\right) - F_k\left(\frac{1}{a+1}\right) = P\left(\frac{1}{a+1} < X_k \leq \frac{1}{a}\right) = \mu(I) = P(A_{k+1} = a) = \lim_{n\to\infty} p_k(a, n).$$

□

Izrek 4.10. *V Evklidovem algoritmu se kvocient, ki je enak a, pojavi s približno verjetnostjo*

$$\lg\left(\frac{(a+1)^2}{(a+1)^2-1}\right).$$

Dokaz. Spomnimo se še enkrat izreka 4.7, velja $F_n(x) = \lg(1+x) + \mathcal{O}(2^{-n})$. Če to povežemo z izrekom 4.9, dobimo naslednji izračun:

$$\begin{aligned}
\lim_{n\to\infty} p_k(a,n) &= F_k\left(\frac{1}{a}\right) - F_k\left(\frac{1}{a+1}\right) \\
&= \lg\left(1+\frac{1}{a}\right) + \mathcal{O}(2^{-k}) - \lg\left(1+\frac{1}{a+1}\right) - \mathcal{O}(2^{-k}) \\
&= \lg\left(\frac{\frac{a+1}{a}}{\frac{a+2}{a+1}}\right) \\
&= \lg\left(\frac{(a+1)^2}{a(a+2)}\right) \\
&= \lg\left(\frac{(a+1)^2}{(a+1)^2-1}\right).
\end{aligned}$$

Število k se tukaj izgubi, zato je verjetnost enaka za katerokoli število k. Iz tega sledi, da se kvocient, ki je enak a, pojavi s približno verjetnostjo $\lg\left(\frac{(a+1)^2}{(a+1)^2-1}\right)$. □

Primeri za nekatere od kvocientov so zapisani v tabeli 4.3 (stran 41).

Če seštejemo verjetnosti pojavitev kvocientov 1, 2 in 3, je vsota enaka $0,415038 + 0,169925 + 0,093109 = 0,678072$, kar pomeni, da se kvocienti 1, 2 in 3 pojavijo v približno $67,8\,\%$ vseh primerov kvocientov v Evklidovem algoritmu.

Za primer si oglejmo verižne ulomke za množico vseh ustreznih ulomkov, katerih imenovalec je enak 37.

$\frac{1}{37} = [37]$	$\frac{10}{37} = [3,1,2,3]$	$\frac{19}{37} = [1,1,18]$	$\frac{28}{37} = [1,3,9]$
$\frac{2}{37} = [18,2]$	$\frac{11}{37} = [3,2,1,3]$	$\frac{20}{37} = [1,1,5,1,2]$	$\frac{29}{37} = [1,3,1,1,1,2]$
$\frac{3}{37} = [12,3]$	$\frac{12}{37} = [3,12]$	$\frac{21}{37} = [1,1,3,5]$	$\frac{30}{37} = [1,4,3,2]$
$\frac{4}{37} = [9,4]$	$\frac{13}{37} = [2,1,5,2]$	$\frac{22}{37} = [1,1,2,7]$	$\frac{31}{37} = [1,5,6]$
$\frac{5}{37} = [7,2,2]$	$\frac{14}{37} = [2,1,1,1,4]$	$\frac{23}{37} = [1,1,1,1,1,4]$	$\frac{32}{37} = [1,6,2,2]$
$\frac{6}{37} = [6,6]$	$\frac{15}{37} = [2,2,7]$	$\frac{24}{37} = [1,1,1,5,2]$	$\frac{33}{37} = [1,8,4]$
$\frac{7}{37} = [5,3,2]$	$\frac{16}{37} = [2,3,5]$	$\frac{25}{37} = [1,2,12]$	$\frac{34}{37} = [1,11,3]$
$\frac{8}{37} = [4,1,1,1,2]$	$\frac{17}{37} = [2,5,1,2]$	$\frac{26}{37} = [1,2,2,1,3]$	$\frac{35}{37} = [1,17,2]$
$\frac{9}{37} = [4,9]$	$\frac{18}{37} = [2,18]$	$\frac{27}{37} = [1,2,1,2,3]$	$\frac{36}{37} = [1,36]$

Na primeru lahko opazimo nekatera zanimiva dejstva:

Kvocient	Verjetnost pojavitve kvocienta		
1	$\lg\left(\frac{4}{3}\right)$	=	$0,415038$
2	$\lg\left(\frac{9}{8}\right)$	=	$0,169925$
3	$\lg\left(\frac{16}{15}\right)$	=	$0,093109$
4	$\lg\left(\frac{25}{24}\right)$	=	$0,058894$
5	$\lg\left(\frac{36}{35}\right)$	=	$0,040642$
6	$\lg\left(\frac{49}{48}\right)$	=	$0,029747$
7	$\lg\left(\frac{64}{63}\right)$	=	$0,022720$
8	$\lg\left(\frac{81}{80}\right)$	=	$0,017922$
9	$\lg\left(\frac{100}{99}\right)$	=	$0,014410$
10	$\lg\left(\frac{121}{120}\right)$	=	$0,011973$
100	$\lg\left(\frac{10201}{10200}\right)$	=	$0,000141$
1000	$\lg\left(\frac{1002001}{1002000}\right)$	=	$0,000001$

Tabela 4.3: Verjetnosti pojavitev nekaterih kvocientov v Evklidovem algoritmu.

a) Vrednosti stolpcev na desni strani so povezane z vrednostmi stolpcev na levi strani. Zanimiva povezava je $1 - [x_1, x_2, \ldots, x_n] = [1, x_1 - 1, x_2, \ldots, x_n]$, recimo $1 - [7, 2, 2] = 1 - \frac{5}{37} = \frac{32}{37} = [1, 6, 2, 2]$.

b) Obstaja simetrija med desno in levo stranjo v prvih dveh stolpcih, npr. $[18, 2] \leftrightarrow [2, 18]$, $[4, 1, 1, 1, 2] \leftrightarrow [2, 1, 1, 1, 4]$, ... Če se pojavi $[A_1, A_2, \ldots, A_t]$, se vedno pojavi tudi $[A_t, A_{t-1}, \ldots, A_1]$.

c) Vseh kvocientov je 124, od tega jih je $\frac{44}{124} = 34,48\,\%$ enakih 1, $\frac{29}{124} = 23,39\,\%$ enakih 2 ter $\frac{15}{124} = 12,09\,\%$ enakih 3, skupaj je to $70,96\,\%$ vseh pojavitev, kar približno ustreza verjetnostim, ki smo jih prej izračunali. Upoštevati moramo, da je to primer na majhnem številu (37) in da bodo odstotki porazdelitev natančnejši pri večjih številih.

V tabeli 4.4 (stran 42) lahko vidimo za nekatere izmed števil porazdelitev delnih kvocientov in tako potrdimo izrek 4.9. Vidimo, da večje kot je izbrano število, bolj se približujemo zgoraj izračunanim odstotkom.

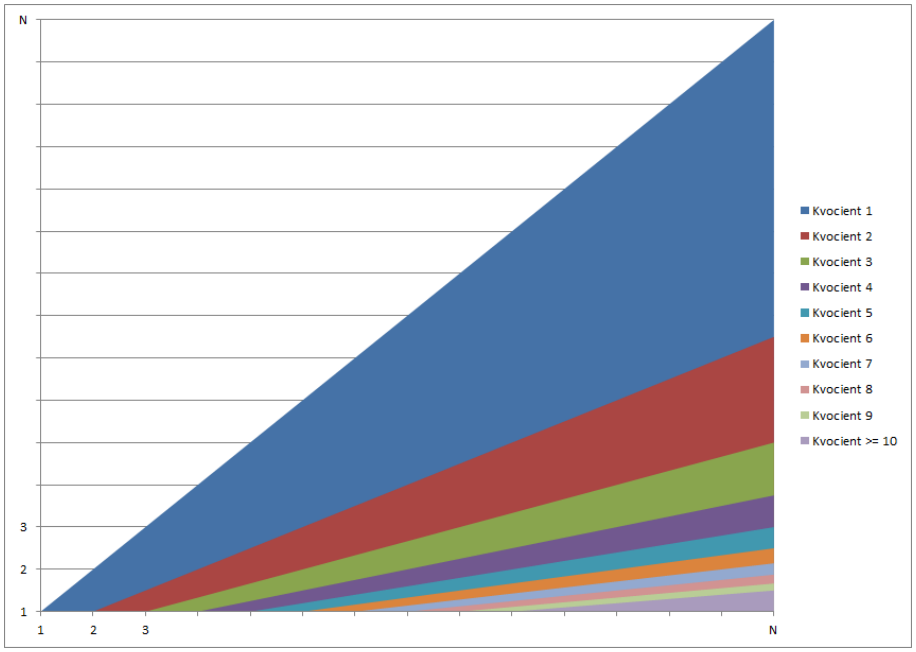
Izbrano število	Število vseh kvocientov	Kvocient 1	Kvocient 2	Kvocient 3
$2^4 = 16$	34	$32,3529\,\%$	$23,5294\,\%$	$17,6471\,\%$
$2^5 = 32$	124	$36,2903\,\%$	$24,1935\,\%$	$12,9032\,\%$
$2^6 = 64$	342	$37,4269\,\%$	$22,8070\,\%$	$13,1579\,\%$
$2^7 = 128$	856	$39,0187\,\%$	$21,9626\,\%$	$11,6822\,\%$
$2^8 = 256$	2010	$38,4577\,\%$	$21,4925\,\%$	$11,4428\,\%$
$2^9 = 512$	4636	$38,8913\,\%$	$21,0095\,\%$	$11,0440\,\%$
$2^{10} = 1024$	10510	$39,4101\,\%$	$20,3045\,\%$	$11,0847\,\%$
$2^{11} = 2048$	23464	$39,7375\,\%$	$20,0136\,\%$	$10,8251\,\%$
$2^{12} = 4096$	51658	$39,6434\,\%$	$19,9853\,\%$	$10,6586\,\%$
$2^{13} = 8192$	112892	$39,7451\,\%$	$19,9040\,\%$	$10,3887\,\%$
$2^{14} = 16384$	244918	$39,8550\,\%$	$19,6956\,\%$	$10,2977\,\%$
$2^{15} = 32768$	528264	$40,0054\,\%$	$19,4834\,\%$	$10,2100\,\%$
$2^{16} = 65536$	1133066	$40,0980\,\%$	$19,3118\,\%$	$10,1532\,\%$
$2^{17} = 131072$	2419356	$40,1887\,\%$	$19,1715\,\%$	$10,0860\,\%$
$2^{18} = 262144$	5144838	$40,2597\,\%$	$19,0437\,\%$	$10,0397\,\%$
$2^{19} = 524288$	10902912	$40,3334\,\%$	$18,9289\,\%$	$9,9955\,\%$
$2^{20} = 1048576$	23031482	$40,3954\,\%$	$18,8284\,\%$	$9,9561\,\%$
$2^{21} = 2097152$	48512572	$40,4482\,\%$	$18,7426\,\%$	$9,9176\,\%$
$2^{22} = 4194304$	101922214	$40,4935\,\%$	$18,6638\,\%$	$9,8866\,\%$
$2^{23} = 8388608$	213647368	$40,5406\,\%$	$18,5887\,\%$	$9,8578\,\%$
$2^{24} = 16777216$	446897362	$40,5827\,\%$	$18,5186\,\%$	$9,8340\,\%$
$2^{25} = 33554432$	932999788	$40,6214\,\%$	$18,4547\,\%$	$9,8115\,\%$
$2^{26} = 67108864$	1944403662	$40,6566\,\%$	$18,3961\,\%$	$9,7912\,\%$

Tabela 4.4: Odstotki porazdelitev delnih kvocientov 1, 2 in 3 za izbrana števila.

Ob tem povejmo, da porazdelitev kvocientov v Evklidovem algoritmu ni enaka porazdelitvi kvocientov

$$q = \left\lfloor \frac{A}{B} \right\rfloor$$

za celi števili A in B, za kateri velja $1 \leq B \leq A \leq N$, kjer je N neko (veliko) naravno število. Za porazdelitev teh kvocientov lahko enostavno narišemo graf (Slika 4.1), iz



Slika 4.1: Porazdelitev kvocientov $q = \left\lfloor \frac{A}{B} \right\rfloor$ za celi števili A in B, za kateri velja $1 \leq B \leq A \leq N$.

katerega je razvidno, da je ploščina za kvocient i, P_{i_N}, enaka

$$P_{i_N} = \frac{\left(\frac{N}{i} - 1\right) \cdot (N - i) - \left(\frac{N}{i+1} - 1\right) \cdot (N - i - 1)}{2},$$

število celih robnih točk na grafu za kvocient i, r_{i_N}, pa je enako

$$r_{i_N} = \left\lfloor \tfrac{N}{i} \right\rfloor + \left(\left\lceil \tfrac{N}{i+1} \right\rceil - \left\lfloor \tfrac{N}{i+2} \right\rfloor - 1 \right).$$

Vseh kvocientov je

$$k_N = 1 + 2 + 3 + \cdots + N = \sum_{k=1}^{N} k.$$

Po Pick–ovem izreku (glej [20]) je ploščina poligona za kvocient i, P_{i_N}, enaka

$$P_{i_N} = n_{i_N} + \frac{r_{i_N}}{2} - 1,$$

kjer n_{i_N} predstavlja število notranjih točk. Tako lahko iz ploščine in števila robnih točk izračunamo število notranjih točk za kvocient i:

$$\begin{aligned} n_{i_N} &= P_{i_N} - \tfrac{r_{i_N}}{2} + 1 \\ &= \frac{\left(\frac{N}{i} - 1\right)(N - i) - \left(\frac{N}{i+1} - 1\right)(N - i - 1)}{2} - \frac{\left\lfloor \frac{N}{i} \right\rfloor + \left(\left\lceil \frac{N}{i+1} \right\rceil - \left\lfloor \frac{N}{i+2} \right\rfloor - 1 \right)}{2} + 1. \end{aligned}$$

Če seštejemo robne in notranje točke ter delimo vsoto s številom vseh točk k_N, dobimo porazdelitve kvocientov. Prav tako lahko izračunamo, kolikšen je odstotek ploščine za določen kvocient, tako da delimo ploščino za kvocient i s ploščino vseh kvocientov. V tabeli 4.5 sta podana ta dva rezultata za kvociente 1, 2 in 3 za nekatera števila. Vidimo, da večja, kot so števila, bolj se ujemajo rezultati Pick–ovega izreka in ploščine grafa in vedno bolj se približujemo temu, da se kvocent 1 pojavlja v $50\,\%$ primerov, kvocient 2 v $16,\overline{6}\,\%$ primerov, kvocient 3 pa v $8,\overline{3}\,\%$ primerov. V tabelah 4.6, 4.7, 4.8 in 4.9 je podano, v koliko odstotkih se pojavi kateri od kvocientov $q = \left\lfloor \frac{A}{B} \right\rfloor$ za celi števili A in B, za kateri velja $1 \leq B \leq A \leq N$. Tudi tu vidimo, da večja, kot so števila, bolj se približujemo temu, da se kvocent 1 pojavlja v $50\,\%$ primerov, kvocient 2 v $16,\overline{6}\,\%$ primerov, kvocient 3 v $8,\overline{3}\,\%$ primerov itd. Porazdelitev je za dovolj velik N enaka

$$\begin{pmatrix} 1 & 2 & 3 & \cdots & N \\ \frac{1}{2} & \frac{1}{6} & \frac{1}{12} & \cdots & \frac{1}{N(N+1)} \end{pmatrix}.$$

Velikost števil v bitih	Kvocient	Porazdelitev kvocientov glede na Pick–ov izrek ter odstotki ploščine za kvociente glede na graf
		50,00101724484326363351999633794650350183865602636679 %
	1	50,00152589054778217998289329414417892034986735728206 %
		16,66730243208223900875332509371296621247030939265860 %
16	2	16,66717528132674411040357346028175187673580420791637 %
		8,33379108691603439024266190599712121905691950094349 %
	3	8,33358762902148459300874350181614511583228842057495 %
		50,00000001552204290845097164002821842600749376555730 %
	1	50,00000002328306436809746832183876108487169520825365 %
		16,66666676367943481512143058553074791850194305837 26 %
32	2	16,66666674427688119085148863978335649873150039379 45 %
		8,33333340318252640600317494015850620476408561891 16 %
	3	8,33333337213844056832068999513229291123513747160 90 %
		50,00000000000000000361400724161834803802857968770488 %
	1	50,00000000000000000542101086242752205733674311926290 %
		16,66666666666666666892542119267813419027534744480837 %
64	2	16,66666666666666666847367028747584068558299864795058 %
		8,33333333333333334959636592061589950304155292076 17 %
	3	8,33333333333333334236835143737920342644562530122 51 %
		50,00000000000000000000000000000000019591572513705 %
	1	50,00000000000000000000000000000000029387358770557 %
		16,66666666666666666666666666666666678911399487732 %
128	2	16,66666666666666666666666666666666676462452923519 %
		8,33333333333333333333333333333333342149540964500 %
	3	8,33333333333333333333333333333333338231226461760 %
		50,00000000000000000000000000000000000000000000000 %
	1	50,00000000000000000000000000000000000000000000000 %
		16,66666666666666666666666666666666666666666666667 %
256	2	16,66666666666666666666666666666666666666666666667 %
		8,33333333333333333333333333333333333333333333333 %
	3	8,33333333333333333333333333333333333333333333333 %

Tabela 4.5: Porazdelitev kvocientov glede na Pick–ov izrek ter odstotki ploščine za kvociente glede na graf.

Velikost števil v bitih:	0	1	2	3	4	5	6
Kvocient							
1	100,0 %	66,$\overline{6}$ %	60,0 %	55,$\overline{5}$ %	52,94117647 %	51,51515151 %	50,76923076 %
2	0 %	33,$\overline{3}$ %	20,0 %	19,$\overline{4}$ %	17,64705882 %	17,23484848 %	16,92307692 %
3	0 %	0 %	10,0 %	8,$\overline{3}$ %	8,82352941 %	8,52272727 %	8,46153846 %
4	0 %	0 %	10,0 %	5,$\overline{5}$ %	5,14705882 %	5,11363636 %	5,09615384 %
5	0 %	0 %	0 %	2,$\overline{7}$ %	3,67647058 %	3,40909090 %	3,36538461 %
6	0 %	0 %	0 %	2,$\overline{7}$ %	2,20588235 %	2,46212121 %	2,40384615 %
7	0 %	0 %	0 %	2,$\overline{7}$ %	2,20588235 %	1,89393939 %	1,82692307 %
8	0 %	0 %	0 %	2,$\overline{7}$ %	1,47058823 %	1,32575757 %	1,39423076 %
9	0 %	0 %	0 %	0 %	0,73529411 %	1,13636363 %	1,10576923 %
od 10 do 99	0 %	0 %	0 %	0 %	5,14705882 %	7,38636363 %	8,65384615 %

Tabela 4.6: Porazdelitev kvocientov v odstotkih za števila velikosti od 0 do 6 bitov.

Velikost števil v bitih:	7	8	9	10	11
Kvocient					
1	50, 38759689 %	50, 19455252 %	50, 09746588 %	50, 04878048 %	50, 02440214 %
2	16, 79990310 %	16, 73151750 %	16, 69940911 %	16, 68292682 %	16, 67481660 %
3	8, 39389534 %	8, 36575875 %	8, 34932383 %	8, 34146341 %	8, 33738447 %
4	5, 03875968 %	5, 01884727 %	5, 00959429 %	5, 00495426 %	5, 00244021 %
5	3, 35513565 %	3, 34691147 %	3, 33972953 %	3, 33650914 %	3, 33494425 %
6	2, 39825581 %	2, 38934824 %	2, 38562987 %	2, 38319359 %	2, 38211665 %
7	1, 80474806 %	1, 79353112 %	1, 78941276 %	1, 78753810 %	1, 78659940 %
8	1, 39292635 %	1, 39226653 %	1, 39193469 %	1, 39024390 %	1, 38954024 %
9	1, 12645348 %	1, 11563715 %	1, 11248172 %	1, 11204268 %	1, 11167985 %
od 10 do 99	8, 95106589 %	9, 00109435 %	9, 01407163 %	9, 00724085 %	9, 00382046 %
od 100 do 999	0, 35125968 %	0, 65053501 %	0, 81094663 %	0, 90034298 %	0, 89992450 %
od 1000 do 1024	0 %	0 %	0 %	0, 00476371 %	0, 05233116 %

Tabela 4.7: Porazdelitev kvocientov v odstotkih za števila velikosti od 7 do 11 bitov.

Velikost števil v bitih:	12	13	14	15
Kvocient				
1	50,01220405 %	50,00610277 %	50,00305157 %	50,00152583 %
2	16,67073468 %	16,66870191 %	16,66768385 %	16,66717533 %
3	8,33536734 %	8,33434946 %	8,33384192 %	8,33358757 %
4	5,00121802 %	5,00060968 %	5,00030545 %	5,00015258 %
5	3,33414932 %	3,33373978 %	3,33353647 %	3,33343499 %
6	2,38153012 %	2,38124270 %	2,38109748 %	2,38102504 %
7	1,78615354 %	1,78593351 %	1,78582348 %	1,78576883 %
8	1,38922391 %	1,38905675 %	1,38897315 %	1,38893135 %
9	1,11137913 %	1,11124659 %	1,11117882 %	1,11114493 %
od 10 do 99	9,00219243 %	9,00109206 %	9,00054549 %	9,00027274 %
od 100 do 999	0,89971511 %	0,89988750 %	0,89997148 %	0,90001291 %
od 1000 do 1024	0,07613230 %	0,08803723 %	0,09399078 %	0,09696783 %

Tabela 4.8: Porazdelitev kvocientov v odstotkih za števila velikosti od 12 do 15 bitov.

Velikost števil v bitih:	16	17	18	19
Kvocient				
1	50, 00076292 %	50, 00038146 %	50, 00019073 %	50, 00009536 %
2	16, 66692097 %	16, 66679382 %	16, 66673024 %	16, 66669845 %
3	8, 33346048 %	8, 33339690 %	8, 33336512 %	8, 33334922 %
4	5, 00007628 %	5, 00003814 %	5, 00001907 %	5, 00000953 %
5	3, 33338420 %	3, 33335876 %	3, 33334604 %	3, 33333969 %
6	2, 38098869 %	2, 38097054 %	2, 38096146 %	2, 38095692 %
7	1, 78574155 %	1, 78572791 %	1, 78572109 %	1, 78571769 %
8	1, 38891008 %	1, 38889947 %	1, 38889418 %	1, 38889153 %
9	1, 11112803 %	1, 11111958 %	1, 11111534 %	1, 11111323 %
od 10 do 99	9, 00013683 %	9, 00006856 %	9, 00003429 %	9, 00001716 %
od 100 do 999	0, 90000848 %	0, 90000658 %	0, 90000328 %	0, 90000164 %
od 1000 do 1024	0, 09848143 %	0, 09923821 %	0, 09961909 %	0, 09980953 %

Tabela 4.9: Porazdelitev kvocientov v odstotkih za števila velikosti od 16 do 19 bitov.

Poglavje 5

Lehmerjev algoritem

Že v poglavju o Evklidovem algoritmu smo v izreku 2.5 pokazali, da je časovna zahtevnost Evklidovega algoritma enaka $O\big((\log_2 M)^2\big)$, kjer sta števili A in B naključno porazdeljeni med številoma 1 in M. Zaradi tega Evklidov algoritem ni učinkovit za velika števila. Izrek 4.10 nam pove, da je kvocient

$$q = \left\lfloor \frac{A}{B} \right\rfloor$$

vsakega koraka večinoma majhen (kvocienti 1, 2 in 3 se pojavijo v približno $67,8\,\%$ vseh primerov kvocientov).

Recimo, da imamo številski sistem z osnovo β. Označimo bazo z W in naj bo $W = \beta^p$ za neko pozitivno celo število p. Primeri baze so $W = 2^{32}$ ($\beta = 2$, $p = 32$), to lahko uporabljamo npr. pri 32–bitnih računalnikih, ali pa $W = 1000$ ($\beta = 10$, $p = 3$), ta baza je uporabna za računanje "na roke".

Lehmer je opazil, da ker je večina kvocientov majhnih, se lahko izognemo deljenju velikih števil. Velikokrat namreč lahko deljenje velikih števil A in B nadomestimo z deljenjem njunih vodilnih delov, $a_0 = \left\lfloor \frac{A}{\beta^h} \right\rfloor$ in $a_1 = \left\lfloor \frac{B}{\beta^h} \right\rfloor$, pri čemer je h najmanjše možno število, za katerega še velja, da je $a_1 < \beta^p$. Število a_1 predstavlja prvih p števk števila B v sistemu z osnovo β.

Pogosto lahko kvocient q izračunamo kar iz števil a_0 in a_1, torej ni potrebno deliti velikih števil A in B, ampak lahko do kvocienta q pridemo na računsko enostavnejši način. Ko števili A in B nista več veliki, lahko uporabimo procesorski ukaz. Ta sicer pri velikih številih računa počasi, v primeru majhnih števil pa se izvede hitro. Primer za iskanje kvocienta v prvem koraku Lehmerjevega algoritma smo si ogledali že v uvodu (Primer 1.2).

5.1 Ideja algoritma

Ideja Lehmerjevega algoritma je, da s čim manj računanja poiščemo števili q' in q'', za kateri velja bodisi $q' \leq q \leq q''$ bodisi $q'' \leq q \leq q'$. Če velja $q' = q''$, potem smo našli kvocient q. Lehmer je števili q' in q'' definiral v prvem koraku algoritma kot

$$q' = \left\lfloor \frac{a_0 + 1}{a_1} \right\rfloor \quad \text{in} \quad q'' = \left\lfloor \frac{a_0}{a_1 + 1} \right\rfloor . \tag{5.1}$$

Nato v nadaljevanju računamo zaporedja $\{a_i\}$, $\{u_i\}$ in $\{v_i\}$, vsako po principu računanja Evklidovega zaporedja:

$$a_{i+2} = a_i - q_i \cdot a_{i+1}, \quad u_{i+2} = u_i - q_i \cdot u_{i+1}, \quad v_{i+2} = v_i - q_i \cdot v_{i+1},$$

pri čemer so začetne vrednosti zaporedij $\{u_i\}$ in $\{v_i\}$ enake

$$u_0 = 1, \quad u_1 = 0, \quad v_0 = 0, \quad v_1 = 1$$

in je $q_i = q' = q''$. S pomočjo vrednosti v teh zaporedjih računamo q' in q'' po pravilu

$$q' = \left\lfloor \frac{a_i + u_i}{a_{i+1} + u_{i+1}} \right\rfloor \quad \text{in} \quad q'' = \left\lfloor \frac{a_i + v_i}{a_{i+1} + v_{i+1}} \right\rfloor .$$

Enostavno lahko pokažemo, da v prvem koraku algoritma velja $q'' \leq q \leq q'$. Ker je $a_0 = \left\lfloor \frac{A}{\beta^h} \right\rfloor$, je $a_0 \cdot \beta^h \leq A$ in ker je $a_1 = \left\lfloor \frac{B}{\beta^h} \right\rfloor$, je $(a_1+1) \cdot \beta^h > B$. Torej je $q'' = \left\lfloor \frac{a_0}{a_1+1} \right\rfloor \leq q$. Podobno pokažemo še desno stran neenakosti. Da velja

$$\left\lfloor \frac{a_i + v_i}{a_{i+1} + v_{i+1}} \right\rfloor \leq \left\lfloor \frac{r_{i-2}}{r_{i-1}} \right\rfloor \leq \left\lfloor \frac{a_i + u_i}{a_{i+1} + u_{i+1}} \right\rfloor$$

za sodo število i ter

$$\left\lfloor \frac{a_i + u_i}{a_{i+1} + u_{i+1}} \right\rfloor \leq \left\lfloor \frac{r_{i-2}}{r_{i-1}} \right\rfloor \leq \left\lfloor \frac{a_i + v_i}{a_{i+1} + v_{i+1}} \right\rfloor$$

za liho število i, pri čemer je $\{r_i\}$ Evklidovo zaporedje z začetnima členoma $r_{-2} = \mathrm{A}$ in $r_{-1} = B$, bomo kasneje pokazali v trditvi 6.1 (stran 58).

Algoritem deluje tako, da najprej vzame vodilne dele obeh števil in računa zaporedje kvocientov, dokler ni ostanek r_i manjši od baze W. Nato nadaljujemo iskanje največjega skupnega delitelja neposredno z Evklidovim algoritmom, saj deljenje od tu naprej ni več tako draga operacija. Operiramo namreč z majhnimi števili, Evklidov algoritem pa je enostavnejši od Lehmerjevega.

5.2 Primerjava z Evklidovim algoritmom

Podobno kot pri Evklidovem algoritmu tudi pri Lehmerjevem algoritmu računamo zaporedje kvocientov $\{q_i\}$, vendar pri Lehmerjevem algoritmu skušamo izračunati q_i brez računsko zamudnega deljenja r_i z r_{i+1}, namesto tega poiščemo kvocient na način, kot smo ga opisali v poglavju 5.1. Tak način iskanja kvocienta q_i se imenuje *Lehmerjevi koraki* (aritmetika z majhnimi števili), izvajamo pa jih, dokler se q' in q'' ujemata.
Če na i–tem koraku zunanje zanke Lehmerjevega algoritma izvedemo k–krat Lehmerjev korak, smo se izognili k–tim deljenjem velikih števil in s tem računanju $k-2$ elementov Evklidovega zaporedja $\{r_i\}$, ki smo ga definirali v 2.3 (stran 6), za katerega velja:

$$r_{-2} = A, \quad r_{-1} = B, \quad r_{i-2} = \left\lfloor \frac{r_{i-2}}{r_{i-1}} \right\rfloor \cdot r_{i-1} + r_i, \quad \text{za } i \geq 0.$$

Lahko se tudi zgodi, da na i–tem koraku ne izvedemo nobenega Lehmerjevega koraka. V tem primeru moramo deliti dve veliki števili r_i in r_{i+1}.
Ko kvocienta nista več enaka, izračunamo r_{i+k} in r_{i+k+1} in nadaljujemo z naslednjim korakom zunanje zanke Lehmerjevega algoritma.

5.3 Inverz z razširjenim Lehmerjevim algoritmom

V razširjenem Lehmerjevem algoritmu, podobno kot pri razširjenem Evklidovem algoritmu, računamo dodatno še zaporedji $\{s_i\}$ in $\{t_i\}$. Osnovnemu Lehmerjevemu algoritmu dodamo še nekaj vrstic, v katerih preračunavamo ti zaporedji, tako da ohranjamo enakost $s_i \cdot A + t_i \cdot B = r_i$, ki je ključna za izračun inverza. Pri tem velja $s_{-2} = 1$, $s_{-1} = 0$, $t_{-2} = 0$ in $t_{-1} = 1$.
Če algoritem uporabimo za izračun inverza števila B v $\mathbb{Z}_n^*$, potem podobno kot pri razširjenem Evklidovem algoritmu tudi pri razširjenem Lehmerjevem algoritmu ne računamo zaporedja $\{s_i\}$. V tem primeru razširjen Lehmerjev algoritem izvedemo za podatke $A = n$ in B. Velja torej $s \cdot n + t \cdot B \equiv t \cdot B \equiv d \pmod{n}$. Ker sta A in B tuji si števili, je $d = 1$. Število $t \pmod{n}$ je multiplikativni inverz števila B v $\mathbb{Z}_n^*$, saj velja $t \cdot B \equiv 1 \pmod{n}$.

Poglavje 6

Implementacija algoritmov

V tem razdelku bomo opisali implementacijo osnovnega in razširjenega Evklidovega algoritma ter osnovnega in razširjenega Lehmerjevega algoritma. Koda je napisana v programskem jeziku `Python`, saj je lahko berljiva. Za testiranje hitrosti algoritmov in preverjanje porazdelitve kvocientov (Poglavje 8) pa smo uporabili programski jezik `Java`.

6.1 Implementacija Evklidovega algoritma

Najprej bomo opisali implementacijo obeh Evklidovih algoritmov, ki se uporabljata tudi v Lehmerjevih algoritmih, ko je $r_i < W$.

6.1.1 Osnovni Evklidov algoritem

```
def EA(A,B):

    while B > 0:
        q = (int)(A/B);
        r = A - q*B;          A = B;          B = r;

    return A;
```

6.1.2 Razširjen Evklidov algoritem

```
def REA(A,B,s0,s1,t0,t1):
```

```
    while B > 0:
        q = (int)(A/B);

        r = A - q*B;        A = B;        B = r;
        r = s0 - q*s1;      s0 = s1;      s1 = r;
        r = t0 - q*t1;      t0 = t1;      t1 = r;

    return A, s0, t0;
```

6.2 Implementacija Lehmerjevega algoritma

Sedaj bomo opisali implementacijo obeh Lehmerjevih algoritmov, za iskanje največjega skupnega delitelja velikih števil in za iskanje inverza v $\mathbb{Z}_n^*$. Preden zapišemo kodo, omenimo le, katera zaporedja predstavljajo spremenljivke v kodi. Spremenljivki $\mathtt{a}_0$ in $\mathtt{a}_1$ predstavljata zaporedje $\{a_i\}$, spremenljivki $\mathtt{u}_0$ in $\mathtt{u}_1$ zaporedje $\{u_i\}$, spremenljivki $\mathtt{v}_0$ in $\mathtt{v}_1$ pa zaporedje $\{v_i\}$. Spremenljivki `A` in `B` predstavljata zaporedje $\{r_i\}$ v Evklidovem algoritmu. Kvocient q' smo v kodi označili s $\mathtt{q}_0$, kvocient q'' pa s $\mathtt{q}_1$. Edini ukaz, ki ni `Pythonov`, je $\mathtt{h} = \mathtt{B.bitLength}() - \mathtt{p} + 1$, saj v `Pythonu` ni ukaza za pridobitev dolžine števila bitov in smo uporabili kar ukaz iz `Jave`.

6.2.1 Osnovni Lehmerjev algoritem

```
def LA(A,B,p,sistem):
    W = math.pow(sistem,p);

    while B >= W:
        h = B.bitLength() - p + 1;
        a0 = A.shiftRight(h);
        a1 = B.shiftRight(h);

        u0 = 1;     u1 = 0;     v0 = 0;     v1 = 1;

        while a1 + u1 != 0 and a1 + v1 != 0:
            q0 = (int)((a0+u0)/(a1+u1));
            q1 = (int)((a0+v0)/(a1+v1));
```

```
            if q0 != q1:
                break;

            r = a0 - q0*a1;          a0 = a1;          a1 = r;
            r = u0 - q0*u1;          u0 = u1;          u1 = r;
            r = v0 - q0*v1;          v0 = v1;          v1 = r;

        if v0 == 0:
            q = (int)(A/B);
            R = A - q*B;      A = B;      B = R;
        else:
            R = u0*A + v0*B;
            T = u1*A + v1*B;
            A = R;
            B = T;

        if B == 0:
            return A;

    A = EA(A,B);

    return A;
```

Na kratko razložimo, kako deluje algoritem.

Kako dobimo na začetku zanke $\mathtt{a}_0$ in $\mathtt{a}_1$, smo opisali že v začetku poglavja 5. Prav tako smo opisali, kako dobimo prvi vrednosti $\mathtt{q}_0$ in $\mathtt{q}_1$ (enačba (5.1)). Če se ti vrednosti že prvič razlikujeta, zapustimo notranjo zanko. V tem primeru velja $\mathtt{v}_0 = 0$ in izvedemo v bistvu en korak Evklidovega algoritma, da dobimo novi vrednosti za `A` in `B`. Če se to stalno dogaja, je to enako, kot če bi poganjali Evklidov algoritem, le da imamo še vmesne nepotrebne korake, s katerimi ugotovimo, da vrednosti nista enaki. V tem primeru bi bilo bolje, če bi uporabili kar Evklidov algoritem. Kljub temu pa izkušnje kažejo, da se q' in q'' tolikokrat ujemata, da se splača uporabljati Lehmerjev algoritem. Sicer računamo $\mathtt{a}_0$, $\mathtt{a}_1$, $\mathtt{u}_0$, $\mathtt{u}_1$, $\mathtt{v}_0$ in $\mathtt{v}_1$ tako kot pri Evklidovem algoritmu, dokler velja $\mathtt{q}_0 = \mathtt{q}_1$. Ko to ne velja več, izračunamo novi vrednosti za `A` in `B` z enačbami $\mathtt{R} = \mathtt{u}_0 \cdot \mathtt{A} + \mathtt{v}_0 \cdot \mathtt{B}$, $\mathtt{T} = \mathtt{u}_1 \cdot \mathtt{A} + \mathtt{v}_1 \cdot \mathtt{B}$, $\mathtt{A} = \mathtt{R}$ in $\mathtt{B} = \mathtt{T}$, kar ustreza računanju diofantskih enačb.

Če je v nekem koraku `B` $\geq$ `W`, v naslednjem pa `B` $=$ `0`, vrnemo za rezultat vrednost `A`, sicer pa, ko je `B` $<$ `W`, nadaljujemo računanje z Evklidovim algoritmom.

Trditev 6.1. *Vedno velja*

$$\left\lfloor \frac{a_k + v_k}{a_{k+1} + v_{k+1}} \right\rfloor \leq \left\lfloor \frac{r_{k-2}}{r_{k-1}} \right\rfloor \leq \left\lfloor \frac{a_k + u_k}{a_{k+1} + u_{k+1}} \right\rfloor$$

za sodo število k ter

$$\left\lfloor \frac{a_k + u_k}{a_{k+1} + u_{k+1}} \right\rfloor \leq \left\lfloor \frac{r_{k-2}}{r_{k-1}} \right\rfloor \leq \left\lfloor \frac{a_k + v_k}{a_{k+1} + v_{k+1}} \right\rfloor$$

za liho število k.

Dokaz. Že na začetku poglavja o ideji algoritma (5.1) smo pokazali, da velja:

$$a_0 \cdot \beta^h \leq A \qquad \text{in} \quad (a_1 + 1) \cdot \beta^h > B.$$

Podobno lahko pokažemo tudi, da velja

$$(a_0 + 1) \cdot \beta^h > A \quad \text{in} \quad a_1 \cdot \beta^h \leq B.$$

Če združimo te štiri neenakosti, dobimo:

$$q'' = \left\lfloor \frac{a_0}{a_1 + 1} \right\rfloor \leq \left\lfloor \frac{A}{B} \right\rfloor \leq \left\lfloor \frac{a_0 + 1}{a_1} \right\rfloor = q'.$$

Če zapišemo drugače, velja:

$$(a_0 + v_0) \cdot \beta^h \leq r_{-2}, \quad (a_1 + v_1) \cdot \beta^h > r_{-1}, \quad (a_0 + u_0) \cdot \beta^h > r_{-2}, \quad (a_1 + u_1) \cdot \beta^h \leq r_{-1}$$

oziroma

$$\left\lfloor \frac{a_0 + v_0}{a_1 + v_1} \right\rfloor \leq \left\lfloor \frac{r_{-2}}{r_{-1}} \right\rfloor \leq \left\lfloor \frac{a_0 + u_0}{a_1 + u_1} \right\rfloor.$$

Predpostavimo, da velja

$$\begin{aligned} &(a_k + v_k) \cdot \beta^h \leq r_{k-2}, \quad (a_{k+1} + v_{k+1}) \cdot \beta^h > r_{k-1}, \\ &(a_k + u_k) \cdot \beta^h > r_{k-2}, \quad (a_{k+1} + u_{k+1}) \cdot \beta^h \leq r_{k-1}. \end{aligned}$$

To velja za sodo število k. Če člene pri indeksu $k+1$ pomnožimo s $q_k = \left\lfloor \frac{r_{k-2}}{r_{k-1}} \right\rfloor$, dobimo:

$$\begin{aligned} (a_k + v_k) \cdot \beta^h - q_k \cdot \big((a_{k+1} + v_{k+1}) \cdot \beta^h\big) &= \big((a_k - q_k \cdot a_{k+1}) + (v_k - q_k \cdot v_{k+1})\big) \cdot \beta^h \\ &= (a_{k+2} + v_{k+2}) \cdot \beta^h. \end{aligned}$$

Po drugi strani pa velja:

$$(a_k + v_k) \cdot \beta^h - q_k \cdot \left((a_{k+1} + v_{k+1}) \cdot \beta^h\right) \leq r_{k-2} - q_k \cdot r_{k-1} = r_k.$$

Torej je $(a_{k+2} + v_{k+2}) \cdot \beta^h \leq r_k$. S podobnim razmislekom dobimo $(a_{k+2} + u_{k+2}) \cdot \beta^h > r_k$. Zaradi tega velja

$$\left\lfloor \frac{a_{k+1} + v_{k+1}}{a_{k+2} + v_{k+2}} \right\rfloor \geq \left\lfloor \frac{r_{k-1}}{r_k} \right\rfloor \geq \left\lfloor \frac{a_{k+1} + u_{k+1}}{a_{k+2} + u_{k+2}} \right\rfloor.$$

Število $k+1$ je liho in trditev je s tem dokazana. □

Trditev 6.2. *Lehmerjev algoritem se konča po končnem številu korakov in vrne pravilen rezultat.*

Skica dokaza. Števili `A` in `B` sta v algoritmu nosilca dveh zaporednih členov Evklidovega zaporedja in ju na začetku ustrezno inicializiramo. Ko je $\mathtt{B} < \mathtt{W}$, nadaljujemo računanje z Evklidovim algoritmom, za katerega vemo, da se konča po končnem številu korakov in vrne pravilen rezultat.
Sedaj moramo pokazati le še pravilnost notranje zanke `while` $\mathtt{B} \geq \mathtt{W}$. Večina vrstic se ujema z Evklidovim algoritmom. Če že v prvem Lehmerjevem koraku velja $\mathtt{q}_0 \neq \mathtt{q}_1$, potem velja $\mathtt{v}_0 = 0$ in dobimo novi `A` in `B` tako, da pravzaprav izvedemo en korak Evklidovega algoritma. Ker izvedemo en korak Evklidovega algoritma vemo, da sta števili `A` in `B` pravilni.
Če velja $\mathtt{q}_0 = \mathtt{q}_1$, potem računamo $\mathtt{a}_0$, $\mathtt{a}_1$, $\mathtt{u}_0$, $\mathtt{u}_1$, $\mathtt{v}_0$ in $\mathtt{v}_1$ tako kot pri razširjenem Evklidovem algoritmu, torej so pravilni. Ko pridemo do $\mathtt{q}_0 \neq \mathtt{q}_1$, velja $\mathtt{v}_0 \neq 0$ in zato računamo $\mathtt{R} = \mathtt{u}_0 \cdot \mathtt{A} + \mathtt{v}_0 \cdot \mathtt{B}$, $\mathtt{T} = \mathtt{u}_1 \cdot \mathtt{A} + \mathtt{v}_1 \cdot \mathtt{B}$, $\mathtt{A} = \mathtt{R}$ in $\mathtt{B} = \mathtt{T}$.
Če se zgodi, da je v nekem koraku $\mathtt{B} \geq \mathtt{W}$, v naslednjem pa $\mathtt{B} = 0$, potem je vrednost `A` iskana rešitev in ni potrebno nadaljevati z Evklidovim algoritmom.

□

6.2.2 Razširjen Lehmerjev algoritem

```
def RLA(A,B,p,sistem):
    W = math.pow(sistem,p);
    s0 = 1;     s1 = 0;     t0 = 0;     t1 = 1;

    while B >= W:
```

```
h = B.bitLength() - p + 1;
a0 = A.shiftRight(h);
a1 = B.shiftRight(h);

u0 = 1;    u1 = 0;    v0 = 0;    v1 = 1;

while a1 + u1 != 0 and a1 + v1 != 0:
    q0 = (int)((a0+u0)/(a1+u1));
    q1 = (int)((a0+v0)/(a1+v1));

    if q0 != q1:
        break;

    r = a0 - q0*a1;   a0 = a1;  a1 = r;
    r = u0 - q0*u1;   u0 = u1;  u1 = r;
    r = v0 - q0*v1;   v0 = v1;  v1 = r;

if v0 == 0:
    q = (int)(A/B);
    R = A - q*B;    A = B;    B = R;
    R = s0 - q*s1;  s0 = s1;  s1 = R;
    R = t0 - q*t1;  t0 = t1;  t1 = R;
else:
    R = u0*A + v0*B;
    T = u1*A + v1*B;
    A = R;
    B = T;

    R = u0*s0 + v0*s1;
    T = u1*s0 + v1*s1;
    s0 = R;
    s1 = T;

    R = u0*t0 + v0*t1;
    T = u1*t0 + v1*t1;
```

```
            t0 = R;
            t1 = T;

        if B == 0:
            return A,s0,t0;

    A, s0, t0 = REA(A,B,s0,s1,t0,t1);

    return A, s0, t0;
```

Trditev 6.3. *Razširjen Lehmerjev algoritem se konča po končnem številu korakov in vrne pravilen rezultat.*

Skica dokaza. Pomagamo si z idejo dokaza osnovnega Lehmerjevega algoritma, ki smo jo ravnokar razložili, ter dejstvom, da se razširjen Evklidov algoritem konča po končnem številu korakov in vrne pravilen rezultat.
Ker v razširjenem Lehmerjevem algoritmu dodamo le vrstice za računanje zaporedij $\{s_i\}$ in $\{t_i\}$, se računanje največjega skupnega delitelja ne spreminja. Računanje teh dveh zaporedij je podobno računanju zaporedja $\{r_i\}$. Kot smo pokazali že pri osnovnem Lehmerjevem algoritmu, se algoritem konča po končnem številu korakov in vrne pravilen rezultat za največji skupni delitelj. Ker se zaporedji $\{s_i\}$ in $\{t_i\}$ računata podobno kot zaporedje $\{r_i\}$ in ker se tudi pri razširjenem Evklidovem algoritmu računata po enakem principu, je to dovolj, da vidimo, da sta rezultata $\mathtt{s}_0$ in $\mathtt{t}_0$ pravilna. □

6.3 Časovna zahtevnost

Trditev 6.4. *Če sta A in B slučajno izbrani naravni števili, manjši od M, potem je časovna zahtevnost Lehmerjevega osnovnega in razširjenega algoritma enaka $O\big((\log_2 M)^2\big)$.*

Dokaz. Ker je večina kvocientov vsakega koraka deljenja pri iskanju največjega skupnega delitelja dveh naključnih števil oz. pri iskanju inverza števila v $\mathbb{Z}_n^*$ majhna, lahko q_i v večini primerov računamo z Lehmerjevimi koraki, ki niso draga operacija. Če q_i ne moremo dobiti na tak način, izvedemo en korak Evklidovega algoritma. Računanje, ki ni del Evklidovega algoritma, z njim pa ugotovimo, da moramo uporabiti Evklidov algoritem, je enostavneje od samega Evklidovega algoritma, zato asimptotično časovna zahtevnost Lehmerjevega algoritma ne more biti slabša od časovne zahtevnosti Evklidovega algoritma, ki je $O\big((\log_2 M)^2\big)$.

Pri razširjenem Lehmerjevem algoritmu dodamo le nekaj vrstic, ki ne vplivajo na časovno zahtevnost, zato lahko uporabimo podoben razmislek. □

Poglavje 7

Primeri računanja

V tem poglavju si bomo ogledali primere za iskanje največjega skupnega delitelja ter za iskanje inverza z osnovnim in razširjenim Evklidovim ter osnovnim in razširjenim Lehmerjevim algoritmom. Pri obeh algoritmih bomo uporabili enaka števila, da bo primerjava lažja. Za Lehmerjev algoritem bomo naredili primere za različne baze W, da bomo lahko primerjali razliko pri računanju.

7.1 Iskanje največjega skupnega delitelja

Oglejmo si primera iskanja največjega skupnega delitelja za števili $A = 204435$ in $B = 12345$.

Primer 7.1.

Evklidov algoritem na številih $A = 204435$ in $B = 12345$:

A	B	q
204435	12345	16
12345	6915	1
6915	5430	1
5430	1485	3
1485	975	1
975	510	1
510	465	1
465	45	10
45	15	3
15	0	

♢

Primer 7.2.

Lehmerjev algoritem na številih $A = 204435$ in $B = 12345$, baza je $W = 1000$:

A	B
204435	12345

a_0	a_1	u_0	u_1	v_0	v_1	q'	q''
2044	123	1	0	0	1	16	16
123	76	0	1	1	-16	1	2

$q' \neq q''$ in $v_0 \neq 0 \Rightarrow$ $R = u_0 \cdot A + v_0 \cdot B = 0 \cdot 204435 + 1 \cdot 12345 = 12345$

$T = u_1 \cdot A + v_1 \cdot B = 1 \cdot 204435 + (-16) \cdot 12345 = 6915$

$A = R = 12345$ in $B = T = 6915$

A	B
12345	6915

a_0	a_1	u_0	u_1	v_0	v_1	q'	q''
1234	691	1	0	0	1	1	1
691	543	0	1	1	-1	1	1
543	148	1	-1	-1	2	3	3
148	99	-1	4	2	-7	1	1
99	49	4	-5	-7	9	2	1

$q' \neq q''$ in $v_0 \neq 0 \Rightarrow$ $R = u_0 \cdot A + v_0 \cdot B = 4 \cdot 12345 + (-7) \cdot 6915 = 975$

$T = u_1 \cdot A + v_1 \cdot B = (-5) \cdot 12345 + 9 \cdot 6915 = 510$

$A = R = 975$ in $B = T = 510$

Ker je $B = 510 < 1000 = W$, nadaljujemo z Evklidovim algoritmom:

A	B	q
975	510	1
510	465	1
465	45	10
45	15	3
15	0	

$\diamondsuit$

Sedaj si oglejmo primere iskanja največjega skupnega delitelja za števili $A = 204434775$ in $B = 54157500$. Tukaj bomo Lehmerjev algoritem uporabili na dveh različnih bazah, $W = 1000$ in $W = 100000$. Opazimo, da kjer je manjša baza, je število Lehmerjevih korakov večje, v tem primeru dobimo največji skupni delitelj brez, da sploh pridemo do Evklidovega algoritma. Kjer je baza večja, je število teh korakov manjše, na koncu pa tudi nadaljujemo z Evklidovim algoritmom, saj je največji skupni delitelj manjši od baze.

Primer 7.3.

Evklidov algoritem na številih $A = 204434775$ in $B = 54157500$:

A	B	q
204434775	54157500	3
54157500	41962275	1
41962275	12195225	3
12195225	5376600	2
5376600	1442025	3
1442025	1050525	1
1050525	391500	2
391500	267525	1
267525	123975	2
123975	19575	6
19575	6525	3
6525	0	

♢

Primer 7.4.

Lehmerjev algoritem na številih $A = 204434775$ in $B = 54157500$, baza je $W = 1000$:

A	B
204434775	54157500

a_0	a_1	u_0	u_1	v_0	v_1	q'	q''
2044	541	1	0	0	1	3	3
541	421	0	1	1	-3	1	1
421	120	1	-1	-3	4	3	3
120	61	-1	4	4	-15	1	2

$q' \neq q''$ in $v_0 \neq 0 \Rightarrow$

$R = u_0 \cdot A + v_0 \cdot B = 12195225$

$T = u_1 \cdot A + v_1 \cdot B = 5376600$

$A = R = 12195225$ in $B = T = 5376600$

A	B
12195225	5376600

a_0	a_1	u_0	u_1	v_0	v_1	q'	q''
1219	537	1	0	0	1	2	2
537	145	0	1	1	-2	3	3
145	102	1	-3	-2	7	1	1

102	43	-3	4	7	-9	2	3

$q' \neq q''$ in $v_0 \neq 0 \Rightarrow$ $R = u_0 \cdot A + v_0 \cdot B = 1050525$

$T = u_1 \cdot A + v_1 \cdot B = 391500$

$A = R = 1050525$ in $B = T = 391500$

A	B
1050525	391500

a_0	a_1	u_0	u_1	v_0	v_1	q'	q''
1050	391	1	0	0	1	2	2
391	268	0	1	1	-2	1	1
268	123	1	-1	-2	3	2	2
123	22	-1	3	3	-8	4	9

$q' \neq q''$ in $v_0 \neq 0 \Rightarrow$ $R = u_0 \cdot A + v_0 \cdot B = 123975$

$T = u_1 \cdot A + v_1 \cdot B = 19575$

$A = R = 123975$ in $B = T = 19575$

A	B
123975	19575

a_0	a_1	u_0	u_1	v_0	v_1	q'	q''
1239	195	1	0	0	1	6	6
195	69	0	1	1	-6	2	3

$q' \neq q''$ in $v_0 \neq 0 \Rightarrow$ $R = u_0 \cdot A + v_0 \cdot B = 19575$

$T = u_1 \cdot A + v_1 \cdot B = 6525$

$A = R = 19575$ in $B = T = 6525$

A	B
19575	6525

a_0	a_1	u_0	u_1	v_0	v_1	q'	q''
1957	652	1	0	0	1	3	2

$q' \neq q''$ in $v_0 = 0 \Rightarrow$ $R = A - \left\lfloor \frac{A}{B} \right\rfloor \cdot B = 19575 - 3 \cdot 6525 = 0$

$A = B = \mathbf{6525}$ in $B = R = 0$ ♢

Primer 7.5.

Lehmerjev algoritem na številih $A = 204434775$ in $B = 54157500$, baza je $W = 100000$:

A	B
204434775	54157500

a_0	a_1	u_0	u_1	v_0	v_1	q'	q''
204434	54157	1	0	0	1	3	3
54157	41963	0	1	1	-3	1	1
41963	12194	1	-1	-3	4	3	3
12194	5381	-1	4	4	-15	2	2
5381	1432	4	-9	-15	34	3	3
1432	1085	-9	31	34	-117	1	1
1085	347	31	-40	-117	151	3	1

$q' \neq q''$ in $v_0 \neq 0 \Rightarrow$ $R = u_0 \cdot A + v_0 \cdot B = 1050525$

$T = u_1 \cdot A + v_1 \cdot B = 391500$

$A = R = 1050525$ in $B = T = 391500$

A	B
1050525	391500

a_0	a_1	u_0	u_1	v_0	v_1	q'	q''
105052	39150	1	0	0	1	2	2
39150	26752	0	1	1	-2	1	1
26752	12398	1	-1	-2	3	2	2
12398	1956	-1	3	3	-8	6	6
1956	662	3	-19	-8	51	3	2

$q' \neq q''$ in $v_0 \neq 0 \Rightarrow$ $R = u_0 \cdot A + v_0 \cdot B = 19575$

$T = u_1 \cdot A + v_1 \cdot B = 6525$

$A = R = 19575$ in $B = T = 6525$

Ker je $B = 6525 < 100000 = W$, nadaljujemo z Evklidovim algoritmom:

A	B	q
19575	6525	3
6525	0	

♢

7.2 Iskanje inverza

Oglejmo si še, kako z razširjenim Evklidovim in Lehmerjevim algoritmom izračunamo inverz števila v $\mathbb{Z}_n^*$. Za primer vzemimo števili $A = 123975$ in $B = 19576$. Torej bomo računali inverz števila 19576 v $\mathbb{Z}_{123975}^*$.

Primer 7.6.

Evklidov algoritem na številih $A = 123975$ in $B = 19576$:

A	B	s_0	s_1	t_0	t_1	q
123975	19576	1	0	0	1	6
19576	6519	0	1	1	-6	3
6519	19	1	-3	-6	19	343
19	2	-3	1030	19	-6523	9
2	1	1030	-9273	-6523	58726	2
1	0	-9273	19576	**58726**	-123975	

♢

Primer 7.7.

Lehmerjev algoritem na številih $A = 123975$ in $B = 19576$, baza je $W = 1000$:

A	B
123975	19576

a_0	a_1	s_0	s_1	t_0	t_1	u_0	u_1	v_0	v_1	q'	q''
1239	195	1	0	0	1	1	0	0	1	6	6
195	69	1	0	0	1	0	1	1	-6	2	3

$q' \neq q''$ in $v_0 \neq 0 \Rightarrow$

$R = u_0 \cdot A + v_0 \cdot B = 0 \cdot 123975 + 1 \cdot 19576 = 19576$

$T = u_1 \cdot A + v_1 \cdot B = 1 \cdot 123975 + (-6) \cdot 19576 = 6519$

$A = R = 19576$ in $B = T = 6519$

$R = u_0 \cdot s_0 + v_0 \cdot s_1 = 0 \cdot 1 + 1 \cdot 0 = 0$

$T = u_1 \cdot s_0 + v_1 \cdot s_1 = 1 \cdot 1 + (-6) \cdot 0 = 1$

$s_0 = R = 0$ in $s_1 = T = 1$

$R = u_0 \cdot t_0 + v_0 \cdot t_1 = 0 \cdot 0 + 1 \cdot 1 = 1$

$T = u_1 \cdot t_0 + v_1 \cdot t_1 = 1 \cdot 0 + (-6) \cdot 1 = -6$

$t_0 = R = 1$ in $t_1 = T = -6$

A	B
19576	6519

a_0	a_1	s_0	s_1	t_0	t_1	u_0	u_1	v_0	v_1	q'	q''
1957	651	0	1	1	-6	1	0	0	1	3	3
651	4	0	1	1	-6	0	1	1	-3	130	652

$q' \neq q''$ in $v_0 \neq 0 \Rightarrow$

$R = u_0 \cdot A + v_0 \cdot B = 0 \cdot 19576 + 1 \cdot 6519 = 6519$

$T = u_1 \cdot A + v_1 \cdot B = 1 \cdot 19576 + (-3) \cdot 6519 = 19$

$A = R = 6519$ in $B = T = 19$

$R = u_0 \cdot s_0 + v_0 \cdot s_1 = 0 \cdot 0 + 1 \cdot 1 = 1$

$T = u_1 \cdot s_0 + v_1 \cdot s_1 = 1 \cdot 0 + (-3) \cdot 1 = -3$

$s_0 = R = 1$ in $s_1 = T = -3$

$$R = u_0 \cdot t_0 + v_0 \cdot t_1 = 0 \cdot 1 + 1 \cdot (-6) = -6$$
$$T = u_1 \cdot t_0 + v_1 \cdot t_1 = 1 \cdot 1 + (-3) \cdot (-6) = 19$$
$$t_0 = R = -6 \text{ in } t_1 = T = 19$$

Ker je $B = 19 < 1000 = W$, nadaljujemo z Evklidovim algoritmom:

A	B	s_0	s_1	t_0	t_1	q
6519	19	1	-3	-6	19	343
19	2	-3	1030	19	-6523	9
2	1	1030	-9273	-6523	58726	2
1	0	-9273	19576	**58726**	-123975	

♢

Preverimo, če je rezultat pravilen: $t_0 \cdot B = 58726 \cdot 19576 = 1149620176 \equiv 1 \pmod{123975}$. Res je, t_0 je inverz števila B po modulu A.

Poglavje 8

Testiranje

V tem poglavju bomo na naključno izbranih številih preverili, v koliko odstotkih je Lehmerjev algoritem hitrejši od Evklidovega. Nato bomo na naključno izbranih številih preverili, ali se porazdelitve delnih kvocientov v Evklidovem algoritmu ujemajo s prej dokazanimi. Prav tako bomo za naključno izbrane pare števil preverili teoretično izračunano porazdelitev njihovih kvocientov še eksperimentalno.

8.1 Primerjava časov

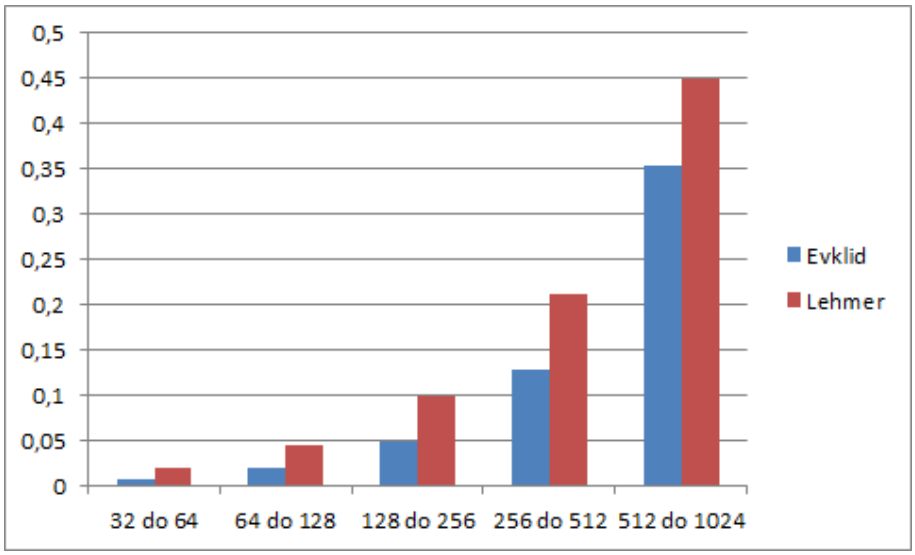
V spodnjih tabelah in grafih so rezultati testiranja primerjave časov. Testirali smo od majhnih števil dalje. Začeli smo z naključnimi števili velikosti med 32 in 64 biti, nato pa povečevali število bitov za faktor 2 in testirali na naključnih številih velikosti trenutne dolžine bitov. Rezultati naše implementacije algoritmov so pokazali, da je Evklidov algoritem hitrejši tam nekje do števil dolžine 1024 bitov, nato pa postane Lehmerjev algoritem hitrejši. V prvi tabeli (Tabela 8.1) in na prvem grafu (Slika 8.1) so rezultati testiranja na številih, kjer je Evklidov algoritem še hitrejši, v drugi (Tabela 8.2) in na drugem ter tretjem grafu (Sliki 8.2 in 8.3) pa rezultati, ko je hitrejši Lehmerjev algoritem.

V prvi tabeli rezultati kažejo povprečno hitrost pri trenutnem izboru bitov, za koliko odstotkov je Evklidov algoritem hitrejši od Lehmerjevega ter za kolikokrat hitreši je. V stolpcu "Število ponovitev" vidimo, kolikokrat smo ponovili algoritem za določeno velikost bitov (vsakič smo izbrali naključna števila). Ko postane Lehmerjev algoritem hitrejši, v drugi tabeli vrnemo rezultate povprečne hitrosti pri trenutnem izboru bitov, za koliko odstotkov je Lehmerjev algoritem hitrejši od Evklidovega ter za kolikokrat hitreši je. Vidimo torej, da večja kot so števila, hitrejši je Lehmerjev algoritem v primerjavi z Evklidovim.

Čemu pa sploh potrebujemo tako velika števila? Oglejmo si sistem RSA, ki ga uporabljamo za faktorizacijo celih števil v kriptografiji.
Na začetku generiramo javni ključ (e, n) in zasebni ključ (d, p, q), tako da:

- izberemo praštevili p in q ter izračunamo modul $n = pq$,
- izračunamo šifrirni eksponent e, tako da je $\gcd(e, \varphi(n)) = 1$, pri čemer je $\varphi(n)$ Eulerjeva funkcija, definirana z $\varphi(n) = \left|\{x \in \mathbb{N}; x < n \text{ in } \gcd(x, n) = 1\}\right|$,
- izračunamo odšifrirni eksponent d iz kongruence $ed \equiv 1 \pmod{\varphi(n)}$ z uporabo razširjenega Evklidovega algoritma.

Glede na priporočila iz leta 2013 (glej [3]), potrebujemo za dolgoročno varnost pri RSA šifriranju 15360–bitne ključe. Če pogledamo v spodnjo tabelo vidimo, da je pri tej velikosti bitov Lehmerjev algoritem za okoli $78,07\,\%$ hitrejši od Evklidovega. Torej nam v tem primeru uporaba razširjenega Lehmerjevega algoritma pride še kako prav.



Slika 8.1: Primerjava časov, ko je Evklidov algoritem hitrejši.

8.2 Porazdelitev kvocientov v Evklidovem algoritmu

Oglejmo si sedaj porazdelitev delnih kvocientov v Evklidovem algoritmu. Tukaj smo testirali podobno kot pri primerjavi časov, le da namesto da bi testirali na naključnih številih dolžine med $\frac{i}{2}$ in i bitov, smo testirali na naključnih številih dolžine med 0 in i bitov. Začeli smo z naključnimi števili velikosti do 64 bitov, nato pa povečevali število bitov za faktor 2 in testirali na naključnih številih velikosti do trenutne dolžine bitov.

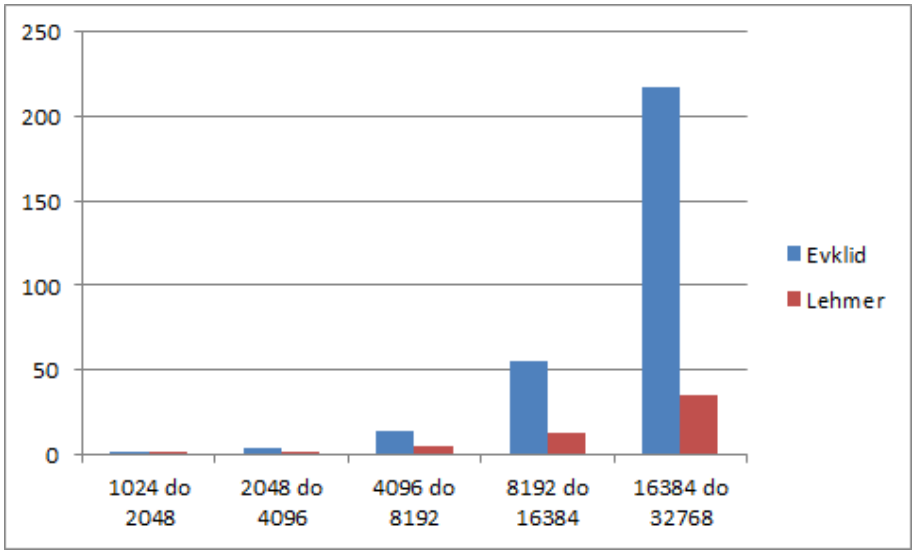
Velikost števil v bitih	Število ponovitev	Povprečni čas Evklidov algoritem	Povprečni čas Lehmerjev algoritem	Koliko je v povprečju hitrejši	Za kolikokrat je v povprečju hitrejši
32 do 64	1048576	0.0079641 ms	0.0192032 ms	58,53 %	2,41–krat
64 do 128	524288	0.0202293 ms	0.0452003 ms	55,25 %	2,23–krat
128 do 256	262144	0.0496101 ms	0.0994911 ms	50,14 %	2,01–krat
256 do 512	131072	0.1279983 ms	0.2115784 ms	39,50 %	1,65–krat
512 do 1024	65536	0.3531952 ms	0.4488525 ms	21,31 %	1,27–krat

Tabela 8.1: Rezultati testiranja, ko je Evklidov algoritem hitrejši.

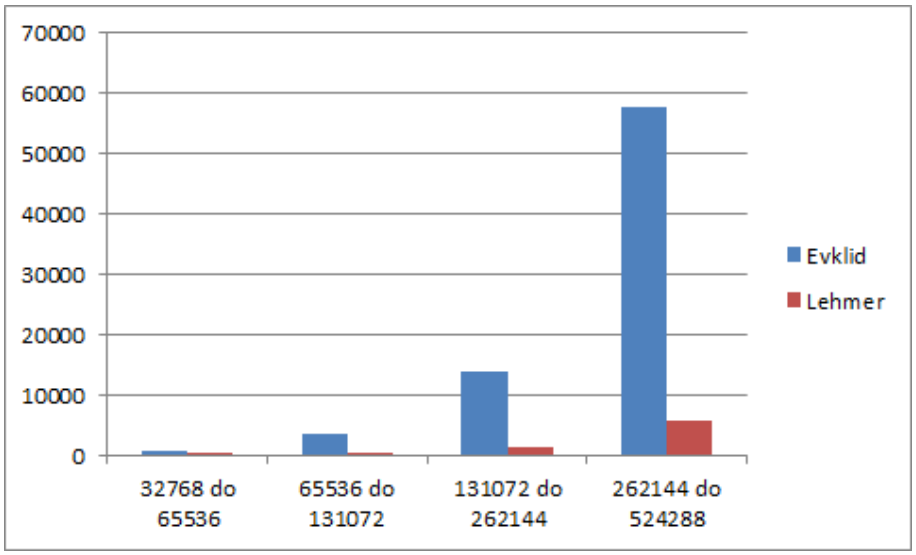
Število ponovitev je bilo enako kot pri testiranju primerjave časov. Rezultati se nahajajo v tabelah 8.3 in 8.4. Vidimo, da velja, kar smo na začetku zapisali o pogostosti pojavitev

Velikost števil v bitih	Število ponovitev	Povprečni čas Evklidov algoritem	Povprečni čas Lehmerjev algoritem	Koliko je v povprečju hitrejši	Za kolikokrat je v povprečju hitrejši
1024 do 2048	32768	1.123 ms	0.941 ms	16, 19 %	1, 19–krat
2048 do 4096	16384	3.853 ms	2.036 ms	47, 17 %	1, 89–krat
4096 do 8192	8192	14.200 ms	4.692 ms	66, 95 %	3, 03–krat
8192 do 16384	4096	55.553 ms	12.181 ms	78, 07 %	4, 56–krat
16384 do 32768	2048	217.703 ms	35.461 ms	83, 71 %	6, 14–krat
32768 do 65536	1024	856.215 ms	113.588 ms	86, 73 %	7, 54–krat
65536 do 131072	512	3397.785 ms	399.668 ms	88, 24 %	8, 50–krat
131072 do 262144	256	13813.629 ms	1497.387 ms	89, 16 %	9, 23–krat
262144 do 524288	128	57725.086 ms	5849.898 ms	89, 87 %	9, 87–krat

Tabela 8.2: Rezultati testiranja, ko je Lehmerjev algoritem hitrejši.



Slika 8.2: Primerjava časov, ko je Lehmerjev algoritem hitrejši - del 1.



Slika 8.3: Primerjava časov, ko je Lehmerjev algoritem hitrejši - del 2.

kvocientov, torej da se kvocienti 1, 2 in 3 pojavijo v približno $67,8\,\%$ vseh primerov kvocientov.

8.3 Kvocienti pri naključno izbranih celih številih

Za konec si oglejmo še porazdelitev kvocientov pri naključno izbranih številih. Tu smo najprej izbrali naključni števili dolžine do 64 bitov, nato smo kot pri prejšnjih testiranjih povečevali število bitov za faktor 2. Pri vsaki dolžini bitov smo testirali na 2^{21} naključnih primerih. Rezultati se nahajajo v tabelah 8.5 in 8.6.

Velikost števil v bitih:	do 64	do 128	do 256	do 512	do 1024	do 2048	do 4096
Delni kvocient							
1	41, 18 %	41, 35 %	41, 42 %	41, 46 %	41, 49 %	41, 48 %	41, 49 %
2	17, 54 %	17, 25 %	17, 12 %	17, 05 %	17, 02 %	17, 01 %	17, 01 %
3	9, 50 %	9, 40 %	9, 36 %	9, 33 %	9, 32 %	9, 31 %	9, 30 %
4	5, 95 %	5, 92 %	5, 90 %	5, 89 %	5, 88 %	5, 88 %	5, 88 %
5	4, 08 %	4, 07 %	4, 07 %	4, 06 %	4, 06 %	4, 06 %	4, 06 %
6	2, 97 %	2, 97 %	2, 97 %	2, 97 %	2, 97 %	2, 97 %	2, 97 %
7	2, 26 %	2, 26 %	2, 26 %	2, 27 %	2, 27 %	2, 27 %	2, 27 %
8	1, 77 %	1, 78 %	1, 78 %	1, 79 %	1, 79 %	1, 79 %	1, 79 %
9	1, 43 %	1, 44 %	1, 44 %	1, 44 %	1, 44 %	1, 44 %	1, 45 %
od 10 do 99	11, 96 %	12, 14 %	12, 21 %	12, 27 %	12, 29 %	12, 31 %	12, 31 %
od 100 do 999	1, 18 %	1, 23 %	1, 26 %	1, 27 %	1, 28 %	1, 28 %	1, 29 %
od 1000 do 1024	0, 12 %	0, 13 %	0, 13 %	0, 14 %	0, 14 %	0, 14 %	0, 14 %

Tabela 8.3: Porazdelitev kvocientov v Evklidovem algoritmu v odstotkih - del 1.

Velikost števil v bitih:	do 8192	do 16384	do 32768	do 65536	do 131072	do 262144
Delni kvocient						
1	41,50 %	41,48 %	41,50 %	41,49 %	41,48 %	41,49 %
2	16,99 %	16,99 %	16,99 %	17,00 %	16,98 %	16,99 %
3	9,31 %	9,31 %	9,31 %	9,31 %	9,31 %	9,30 %
4	5,88 %	5,89 %	5,88 %	5,88 %	5,89 %	5,89 %
5	4,06 %	4,06 %	4,06 %	4,06 %	4,06 %	4,06 %
6	2,97 %	2,97 %	2,97 %	2,97 %	2,97 %	2,97 %
7	2,27 %	2,27 %	2,27 %	2,27 %	2,27 %	2,27 %
8	1,78 %	1,79 %	1,79 %	1,79 %	1,79 %	1,79 %
9	1,45 %	1,44 %	1,44 %	1,45 %	1,45 %	1,45 %
od 10 do 99	12,30 %	12,31 %	12,30 %	12,31 %	12,31 %	12,31 %
od 100 do 999	1,29 %	1,29 %	1,29 %	1,28 %	1,29 %	1,28 %
od 1000 do 1024	0,14 %	0,14 %	0,14 %	0,14 %	0,14 %	0,14 %

Tabela 8.4: Porazdelitev kvocientov v Evklidovem algoritmu v odstotkih - del 2.

Velikost števil v bitih:	do 64	do 128	do 256	do 512	do 1024
Kvocient					
1	49, 98717308 %	50, 00066757 %	50, 00901222 %	49, 99570846 %	49, 99680519 %
2	16, 69421195 %	16, 66412353 %	16, 66078567 %	16, 67675971 %	16, 66650772 %
3	8, 34069252 %	8, 33621025 %	8, 33377838 %	8, 32910537 %	8, 33578109 %
4	5, 00211715 %	4, 99544143 %	4, 99329566 %	4, 99916076 %	5, 00192642 %
5	3, 33743095 %	3, 33895683 %	3, 33523750 %	3, 33504676 %	3, 32751274 %
6	2, 38614082 %	2, 37460136 %	2, 38866806 %	2, 38313674 %	2, 38718986 %
7	1, 76753997 %	1, 78699493 %	1, 78079605 %	1, 78427696 %	1, 78518295 %
8	1, 38759613 %	1, 39484405 %	1, 38740539 %	1, 38826370 %	1, 38845443 %
9	1, 10907554 %	1, 10616683 %	1, 11508369 %	1, 11274719 %	1, 11193656 %
od 10 do 99	8, 99477005 %	9, 00292396 %	9, 00206565 %	8, 99376869 %	8, 99872779 %
od 100 do 999	0, 89468955 %	0, 89755058 %	0, 89578628 %	0, 90050697 %	0, 90150833 %
od 1000 do 1024	0, 09856224 %	0, 10151863 %	0, 09808540 %	0, 10151863 %	0, 09846687 %

Tabela 8.5: Porazdelitev kvocientov v odstotkih - del 1.

Velikost števil v bitih:	do 2048	do 4096	do 8192	do 16384	do 32768
Kvocient					
1	49,99608993 %	49,99861717 %	49,99470710 %	50,05145072 %	50,02851486 %
2	16,66908264 %	16,67289733 %	16,68772697 %	16,81332588 %	17,46153831 %
3	8,33330154 %	8,32753181 %	8,31761360 %	8,14604759 %	9,17067527 %
4	5,00216484 %	5,00125885 %	5,01470565 %	5,17163276 %	4,57429885 %
5	3,33018302 %	3,33547592 %	3,33261489 %	3,29599380 %	2,76465415 %
6	2,37836837 %	2,37689018 %	2,38561630 %	2,22029685 %	1,81574821 %
7	1,79018974 %	1,78680419 %	1,76553726 %	1,80950164 %	1,32827758 %
8	1,38750076 %	1,38874053 %	1,40690803 %	1,36680603 %	0,97126960 %
9	1,11169815 %	1,11150741 %	1,10635757 %	1,14421844 %	0,82426071 %
od 10 do 99	9,00092124 %	8,99949073 %	9,00754928 %	8,97049903 %	10,04905700 %
od 100 do 999	0,90060234 %	0,90141296 %	0,88248252 %	0,90899467 %	0,91137886 %
od 1000 do 1024	0,09989738 %	0,09937286 %	0,09818077 %	0,10123252 %	0,10032653 %

Tabela 8.6: Porazdelitev kvocientov v odstotkih - del 2.

Literatura

[1] H.Cohen, A course in computational algebraic number theory, 1996.

[2] G. E. Collins, Lecture notes on arithmetic algorithms, Univerza v Wisconsinu, 1980.

[3] ENISA, Algorithms - Key Size and Parameters Report - 2013 Recommendations, version 1.0 – October 2013.

[4] T. Jebelean, A Double-Digit Lehmer-Euclid Algorithm for Finding the GCD of Long Integers, Journal of Symbolic Computation, volume 19, pages 145–157, 1995.

[5] A. Jurišić, predavanja pri predmetu Kriptografija in računalniška varnost, Ljubljana, Fakulteta za računalništvo in informatiko, 2012.

[6] D. E. Knuth, The art of computer programming, volume 2, Addison-Wesley, 2. izdaja, 1981.

[7] M. Langus, Učinkoviti algoritmi za izvedbo kriptografskih postopkov, seminarska naloga, Ljubljana, Fakulteta za elektrotehniko, 2006.

[8] D. H. Lehmer, Euclid's algorithm for large numbers, American Mathematics Monthly **45**, pages 227–233, 1938.

[9] S. Maksimovič, Učinkovita aritmetika v praštevilskih obsegih, diplomsko delo, Ljubljana, Fakulteta za matematiko in fiziko, 2003.

[10] R. A. Mollin, An Introduction to Cryptography, Second Edition, 2010.

[11] K.H. Paranjape, Some Lectures on Number Theory, Elliptic Curves and Cryptology, 2002, dosegljivo na URL: http://www.imsc.res.in/∼kapil/crypto/notes/main.html.

[12] B. Plestenjak, predavanja pri predmetu Numerična analiza, Ljubljana, Fakulteta za matematiko in fiziko, 2004.

[13] W. Rudin, Principles of Mathematical Analysis, United States of America : McGraw-Hill, 3. izdaja, 1976.

[14] J. Sorenson, An Analysis of Lehmer's Euclidean GCD Algorithm. In *Proc. AMC ISSAC'95 Symp.*, pages 254–258, 1995.

[15] M. Urlep, Analiza Evklidovega algoritma, seminarska naloga pri predmetu Kriptografija, 2005.

[16] I. Vidav, Višja matematika I, Ljubljana : DMFA - založništvo, 2008.

[17] D. Zwillinger, S. Kokoska, CRC Standard Probability and Statistics Tables and Formulae, 2000.

[18] Lehmer's Algorithm - GNU MP 5.0.4, dosegljivo na URL: https://gmplib.org/manual/Lehmer_0027s-Algorithm.html.

[19] Lehmer's Algorithm - GNU MP, The GNU Multiple Precision Arithmetic Library, Edition 4.3.8, 2009, dosegljivo na URL: http://www.scribd.com/doc/56628567/102/Lehmer%E2%80%99s-algorithm.

[20] Pick's theorem, dosegljivo na URL: http://en.wikipedia.org/wiki/Pick's_theorem.